DOCTORAL THESIS

Multi Agent Enhanced Business Intelligence For Localized Automatic Pricing In Grocery Chains

Loebbert, Alexander

Award date:
2012

BOND UNIVERSITY
SCHOOL OF INFORMATION TECHNOLOGY

Multi Agent Enhanced Business Intelligence For Localized Automatic Pricing In Grocery Chains

Alexander Patrick Joseph Loebbert

MIT(Honours), MBA, Int. Dipl. Betriebswirt (IBS)

A dissertation submitted in fulfilment of the requirements of the degree of
Doctor of Philosophy for the School of Information Technology, Bond University.

December 2011

Abstract

Business Intelligence Systems (BI) describe a form of data driven Decision Support Systems (DSS) that integrate a variety of concepts and technologies to gather, store and analyse data. Traditionally the focus of BI is on strategic and tactical decision support by providing decision makers a centralised and holistic view on organisational data. Today businesses are generating increasingly larger amounts of data due to regulatory requirements, business needs and new technologies. Managing and using this data in business decisions can be difficult because of the volume of the data, time pressure and general complexity of today's business problems. In recent years there is a trend to extend BI to an operational level and make BI capabilities available to more workers. In addition to the technological change, business literature suggests the increasing importance of focusing on local market characteristics instead of standardisation across markets. The traditional BI concept does not fully reflect these operational and local requirements and should adapt to this new environment and these requirements to better support businesses in their decision making activities.

Agent and Multi Agent technology is often mentioned as an approach to design and develop flexible and distributed software systems. The technology is used in this research to design the Multi Agent Enhanced Business Intelligence (MAEBI) framework that focuses on distributing decision making capabilities throughout an organisation. Core to the MAEBI framework is the so called Decision Unit (DU) that encapsulates BI functionality with the extension of a Decision Execution (DE) module that allows implementing (changing business process) a decision without human interaction. The agent based design allows embedding a DU in the problem domain to make decisions with a local perspective. Despite the local focus of the MAEBI concept some aspects of the "centralised" BI approach are still maintained.

A prototype, pMAEBI (p=pricing), was implemented in the context of multi store retail pricing. Pricing is an important and complex problem for retailers and it

allows demonstration of some of the capabilities of a MAEBI based system. To evaluate the pMAEBI system a simulation testbed was implemented to analyse the prototype in comparison to a traditional "centralised" system. Simulation results indicate that the pMAEBI managed stores performed better (in terms of profit) than the comparison stores. These results indicate that the MAEBI concept is viable.

Statement of Originality

This thesis represents my own original work towards this research degree and contains no material which has been previously submitted for a degree or diploma at this University or any other institution, except where due acknowledgement is made".

Alexander P. J. Loebbert

Gold Coast, December 2011

Acknowledgements

Firstly, I would like to express my sincerest thanks and appreciation to my supervisor Prof. Gavin Finnie for his continuing support, advice, feedback and patience throughout my candidature. He also deserves thanks and acknowledgment for his advice and guidance in other academic areas, such as teaching and publishing.

I would like to thank Bond University and the School of IT for giving me the opportunity to undertake this research and providing me with the necessary resources and support. I am also very grateful that I was given the opportunity to teach.

I would like to thank all my friends, colleges, students and fellow PhD candidates that I met during my time at Bond University. During uncountable chats, discussions and coffee sessions, they all contributed in some way to my development and sometimes helped me to see things differently. I wish everybody the very best for their future.

Finally yet importantly, I want to thank my parents, Walter and Gabriela, whose patience and continuing support allowed me to do this research.

Publication Arising from this Research

Loebbert, A., & Finnie, G. (2012). *A Multi-Agent Framework for Distributed Business Intelligence Systems.* HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES, Wailea, Maui, Hawaii, USA. (accepted)

Loebbert, A., & Finnie, G. (2012). *Using AOSE to Develop a Local Knowledge Distributed Business Intelligence Solution.* 6th International KES Conference on Agents and Multi-agent Systems – Technologies and Applications, Dubrovnik, Croatia. (submitted)

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Chapter 1

1.1 Introduction

Almost two decades ago, Peter Drucker stressed the evolving importance of knowledge over other economic inputs. In his book 'Managing the Future', he writes: "From now on, the key is knowledge. The world is not becoming labor intensive, not material intensive, not energy intensive, but knowledge intensive" (Drucker, 1993). Drucker's prediction still stands and Davenport (2006) describes how some companies were able to "embrace" technologies that help to manage and analyse data and turn this knowledge into a competitive advantage. He further argues that "analytics" will be an essential part of business success.

Decision Support Systems (DSS) are the type of software systems that help organisations to handle the available data and turn it into information. DSS have a relative long history and represent a core subject area in the Information Systems (IS) discipline (Burstein and Holsapple, 2008). Business Intelligence (BI) is an integrated DSS approach and combines data gathering, data storage and analysis capabilities; BI can be loosely defined as "data-driven DSS" (Negash and Gray, 2008).

Organisations today continue to generate, gather, and store significant amounts of data at an increasing rate. This is driven by regulatory requirements (e.g. Accounting/Tax, Certifications etc.), business needs (e.g. Operations Management, Finance etc.) or technologies like barcodes, RFID and the Internet that make it easier to capture data. This increase in available data has not necessarily led to improved decision making, as software tools have not kept up and it is difficult to put the data into “meaningful and productive” use (Barone, Yu, Won, Jiang, and Mylopoulos, 2010; Sargut and McGrath, 2011).

DSS and BI systems traditionally focus on strategic and tactical decision support, however in recent years we have seen a shift towards DSS and BI at an operational level. In this regard there are a variety of concepts, architectures and ideas suggested to achieve those goals. Examples are Real-Time BI, Embedded BI, Operational BI and Adaptive BI (Azvine, Cui, Nauck, and Majeed, 2006; Barone, et al., 2010; Bucklin, Lehmann, and Little, 1998; Kemper and Baars, 2009; Michalewicz, Schmidt, Michalewicz, and Chiriac, 2007). All these concepts have in common that they focus on combining BI and operational systems to make decision support available on all organisational levels (Kemper and Baars, 2009; Negash and Gray, 2008).

Business has changed significantly over past decades and decision making is much more complex, sometimes even outside human cognitive abilities (Sargut and McGrath, 2011; Vercellis, 2009). IT in general, but in particular DSS/BI systems do support decision makers and the trend towards “BI for the Masses” will extend these support capabilities to more workers across businesses (Negash and Gray, 2008). However, there seems to be an architectural misalignment between BI and business requirements. Companies in the past often followed a strategy of “standardisation” and focused on centralisation to increase efficiency and achieve economies of scale. Competition has become more intense and business literature suggests that for companies to be successful in the future they will have to focus on local market characteristics (Negash and Gray, 2008; Rigby and Vishwanath, 2006; Sargut and McGrath, 2011). BI systems are usually built around a central Data Warehouse (DW). A DW is the source for analysis and decision support activities and is populated through an Extract Transform Load (ETL) process. This ETL process usually runs at pre-defined times and requires time to process and transfer data. This means that data is usually not accessible to decision makers in real-time and is likely to have only limited operational value. High granularity data about local demand characteristics might be lost due to some form of aggregation during the ETL process (D'Souza and White, 2006; Kemper and Baars, 2009; Meredith, O'Donnell, and Arnott, 2008; Negash and Gray, 2008; Trivedi, 2011). This means that the traditional BI approach does not fully support the emerging business needs of operational and local decision support.

This research is concerned with investigating this problem area, in particular by using agent and multi agent technology. Agent and Multi Agent Systems (e.g. Russell, Norvig, Canny, Malik, and Edwards, 2002; Wooldridge, 2001) are often mentioned as a development and architecture approach that has the potential to design and implement intelligent, flexible and adaptive systems. Agent based software design is not yet broadly adapted in mainstream software development, but it was described as "… superior to other technologies and organisations when the environmental situations are highly complex." (Lockemann, 2006, p. 17).

1.2 Application

Complex business environments were mentioned as a driving force to improve BI. One example for a complex environment is pricing and in particular retail pricing. The complexity stems from the fact that pricing is not a 'straight forward' decision. Many pricing concepts and strategies exist but the large number of pricing decisions to be made (e.g. grocery stores stock around 30000 Stock Keeping Unit (SKU) multiplied by the number of stores in chain) and the frequency they have to be repeated (e.g. once a week) still present a challenge to retailers.

Pricing is a major problem for retailers (Bolton, Shankar, and Montoya, 2005; von der Gathen, Daus, and Simon, 2005) as managers still see pricing as art rather than as science and quite often use a "rule of thumb" approach for pricing decisions.

Price is only one of the "4P"s of marketing (Product, Price, Place, Promotion), but in comparison, the factor which is the easiest to change and ultimately connects supply and demand. To support managers, so called Pricing DSS (PDSS) systems have been suggested by academics and are commercially available, but still do not address retailer's needs (Montgomery, 2005; Natter, Reutterer, Mild, and Taudes, 2007). The availability of demand data and DSS/BI capabilities however allow "IT-enhanced pricing strategies" to improve pricing (Dixit, Whipple, Zinkhan, and Gailey, 2007). A trend that retailers have to face is that their centralised, 1-size-fits-all approach does not work as well as it did and this trend will continue. Retailers have to adjust to local market characteristics (Rigby and Vishwanath, 2006).

1.3 Research Questions

The proposed Multi Agent Enhanced Business Intelligence concept (MAEBI) aims to be an "enhanced" form of BI. Merriam-Webster's defines 'enhance' as "to increase or improve in value, quality, desirability, or attractiveness" (Merriam-Webster, 2011). MAEBI should be more valuable to the user (businesses) in certain classes of applications as it aims to overcome some of the shortcomings of the traditional BI approach and supports decision making in complex environments. Complex environments refers to environments where local knowledge and local decision making might lead to better results than a centralised (traditional) approach.

More specifically, this research focuses on decision making in retail chains. The pricing problem is used to demonstrate the proposed MAEBI system. The broad research question is:

"Does the combination of Business Intelligence and Multi Agent Systems provide an advantage compared to centralised Business Intelligence in respect to its applicability to deliver localised decision automation in multi store retail organisations?"

In more detail, the research proposes a business intelligence architecture that utilises multi agent technology to better align BI with the organisational structure and provide decision making capabilities on a local level but still maintaining a corporate system. While local decision making and control is deemed favourable this does not mean that local entities (e.g. store of a chain or franchisee) have complete freedom. Data consolidation on a corporate (HQ) level is still required for strategic and tactical decision making and other corporate activities (e.g. Tax, TQM, Reporting). The proposed architecture will be applied to pricing in a multi store grocery setting to prove feasibility. In this context the term "advantage" in the research question is understood and measured as profit.

1.4 Methodology

In Information Systems Research there is some disagreement about what constitutes research and how it should be executed. Gallupe (2007) writes in this relation, "Current IS research seems more concerned with 'how' the research is conducted rather than 'what' research is conducted and 'why'. Arnott and Pervan (2008) focus on research issues related to decision support systems and one of the problems they identified is that relevance of DSS is often neglected for the sake of rigor.

The methodology of this research is based on the design science research (DSR) paradigm. In particular Hevner et al. (2004) and Hevner & Chatterjee (2010) describe this approach as suitable for IS research. According to the authors DSR seeks solutions to "important and relevant business problems".
In addition to Hevner's DSR guidelines, Peffers et al. (2008) suggest a Design Science Research Process model (Figure 1.1) that covers the entire research project from motivation to communication. This process model was adapted to structure this research.

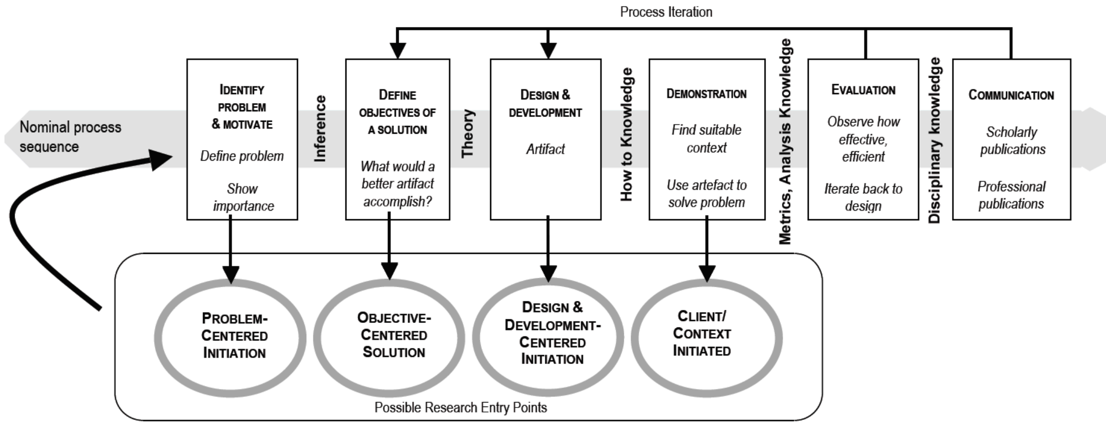

Figure 1.1 Design Science Research Methodology (DSRM) Process Model (Peffers, et al., 2008, p. 14)

To evaluate the proposed MAEBI concept (the artefact), a prototype / testbed is implemented, pMAEBI (p = pricing), that applies the MAEBI framework in the context of retail pricing. This prototype is then tested in a retail simulation. Simulation was chosen as the evaluation approach as it is suggested by Hevner et al. (2004) as part of the general methodology and is also mentioned in agent literature to test new systems (e.g. Theodoropoulos, Minson, Ewald, and Lees, 2009).

The methodology is described in detail in chapter 3.

1.5 Justification of the Research and Contribution

Decision Support Systems are core to the IS discipline and can have significant impact on the performance of an organisation (Arnott and Pervan, 2008; Burstein and Holsapple, 2008). BI represents the current state of the art approach in decision support systems. O'Leary (2008) argues that DSS have to evolve over time to address the changing environment of the systems. The environment of DSS systems consists of the technologies and concepts that are used to develop the systems and the user and its requirements. Decision making has become a more complex process caused by competitive business environments and the need for more efficiency (Hall, 2008). The issue of complexity in the context of business increases according to Sargut & McGrath (2011). They distinguish between complicated business and complex business and write: "It's harder to make sense of things, because the degree of complexity may lie beyond our cognitive limits." and further argue "Making matters worse, our analytic tools haven't kept up.".

In addition to the actual research question there are qualitative insights or contributions of this research. In particular in the area of agent and multi agent technology and the testing of such systems.

Contribution 1 (distribution of localised decision making capabilities):
Decision Support Systems are important systems that find application in virtually all industries and contexts. The general environment has become more complex, in particular interconnected systems and sensors generate more data and good decision making might be out of the reach of our human cognitive limits and even the limits of some software systems.

The proposed MAEBI concept presents a DSS system that builds on what we currently know as BI and aims to make it more dynamic, flexible and adaptable to deliver decision making capabilities throughout an organisation. In particular the MAEBI concept focuses on local decision making and does not follow a centralised approach as traditional BI does.

Contribution 2 (Agent Oriented Software Engineering):
Agent and Multi Agent technology has significant promise to design and implement intelligent and adaptive systems. However advantages over current practices have to be proven and mainstream adaptation is still low. As it turns out, proving that Agents are in fact a better option is difficult. Georgeff (2009) wrote "The Agent

community to date has done a poor job of convincing business and mainstream software engineers of the value of Multi-Agent Systems (MAS) and agent-oriented software engineering."

Winikoff (2009) suggests that documenting MAS/AOSE systems to relevant problems should help to promote the technology and bring experience to the field. DSS systems are surely relevant and applicable to many situations. By using an agent oriented design, this work adds experience in form of an additional documented MAS system and eventually contributes to the body of knowledge.

Contribution 3 (MAS Evaluation and Benchmarking):
Similar to the problem that Winikoff (2009) mentions, that there are not enough documented MAS examples, this can be extended to the testing and evaluation of the systems that are built using the approach. This research contributes in so far, that it adds a new implementation of a testing approach that might help to find future standards and best practices in agent testing.

1.6 Thesis Structure

The structure of the thesis follows the Design Science Research Publication Schema suggested by (Gregor and Hevner, 2011) and follows the DSR process model described by (Peffers, et al., 2008). Both models are explained in detail in chapter 3. Figure 1.2 illustrates the 'flow' and relation of the different aspects of the research, to give the reader a visual overview of the thesis structure.

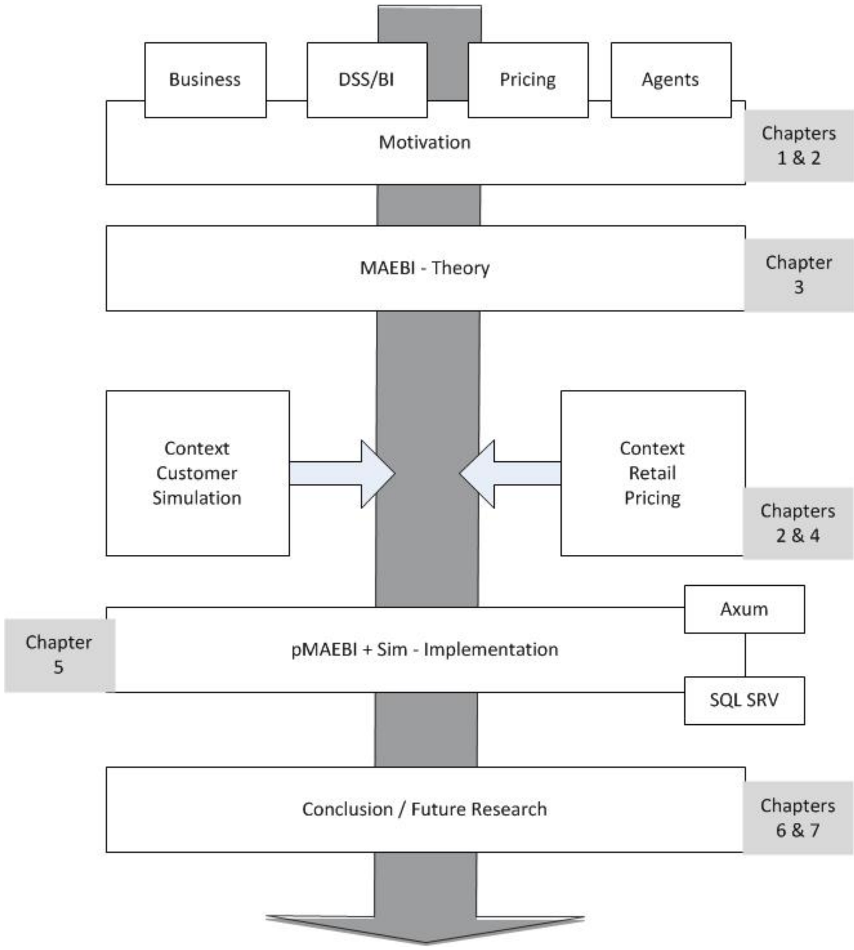

Figure 1.2 - Research Overview

The rest of the thesis is organized as follows:

Chapter 2 presents the literature review. It covers the respective knowledge base for this research, in particular the topics of DSS / BI, Agent and Multi Agent Systems and Retail Pricing.

Chapter 3 describes the development of the methodology for this research.

Chapter 4 describes the design objectives of the proposed system and how those are transformed into the Multi Agent Enhanced Business Intelligent (MAEBI) concept.

Chapter 5 covers the evaluation of the MAEBI concept. The chapter describes the design of the pMAEBI system that applies the MAEBI concept to retail pricing.

Chapter 6 evaluates the design science research process based on Hevner's et al. (2004) DSR guidelines.

Chapter 7 concludes the thesis by summarising the research findings and makes suggestions about future research.

Chapter 2 Literature Review

2.1 Introduction

Decision Support Systems is an important part of Information Systems research and the field continuously grows. The environment of DSS systems, the businesses that use DSS systems and the technologies that are used to design and implement such systems, change continuously and this requires that DSS concepts have to adapt accordingly (Burstein and Holsapple, 2008; O'Leary, 2008). Edmunds & Morris (2000) argued that data is generally available in organisations, yet transforming it into useful information is a challenge. Barone (2010) argues that this problem still exists and that todays Business Imtelligence (BI) systems do not address business needs sufficiently.

The motivation behind this research is to design a new or enhanced version of the "traditional" BI concept. To do so, this literature review covers three main areas that present the basis for this research, Decision Support Systems (DSS) and Business Intelligence (BI), Agent and Multi Agent Systems and Retail Pricing. (In Chapter 3 the research methodology will be introduced in detail. This literature presents the relevant knowledge base in Hevner's Model that is used to develop the methodology.)

DSS and BI are closely related concepts; section 2.2 will define the terms in more depth and how they are used in context of this research. The section will continue with describing what is currently understood as BI. Further problems and current developments are presented that will lead into the research gap that is addressed in the research.

Agent and Multi Agent systems present an interesting set of new technologies to design complex and adaptive systems. It is the motivation behind this research to leverage this technology and its characteristics in BI systems to better align BI with business demands. Section 2.3 will present some key aspects of the technology and why it is relevant.

Section 2.4 presents an overview of retail pricing. The domain of retail pricing is used to implement the proof of concept testbed. Retail pricing is a significant and complex process that retailers have to manage.

2.2 Decision Support Systems & Business Intelligence

2.2.1 Introduction to BI and DSS

Decision Support Systems and Business Intelligence (BI) are two terms that are related, sometimes used interchangeably, sometimes to describe an evolutionary step in software concepts that support decision making. Decision Support Systems have a relatively long history, and go back to the efforts of Gorry & Scott Morton (1971), Anthony (1965) and Simon (1960). Decision support is deemed to be a core research field in information systems (Burstein and Holsapple, 2008).

Despite the history, DSS still lacks a common definition. Arnott & Pervan (2008) argue that in particular in the field of DSS, "practice leads theory", which might be contributing to the problem. Besides, the term "explains itself", thus it can generally be applied to applications that support decision-making. Following this definition would mean that virtually all (business) applications are DSS systems. Turban et al. (2005) distinguish between DSS as an umbrella term that indeed describes all sorts of software systems that support decision making and DSS as an application, that brings together data storage and a model storage that can be accessed by a user.
The term Business Intelligence (BI) was first introduced in the early 1990s (Vercellis, 2009). Howard Dresner, a research fellow at Gartner Research is frequently mentioned to have coined the term. Negash and Gray (2008) call BI a

data-driven DSS that integrates earlier DSS concepts like Executive Information Systems (EIS), Data Warehouse (DW) and Online Analytical Processing (OLAP). Generally DSS systems have the purpose to support the decision making process in an environment. The environment of a DSS (or BI) system consists of the (human) user and the user's requirements (or organisational requirements) and of the technologies and methods that are used to design and implement such systems. DSS systems have to evolve and to reflect the change in their environment (O'Leary, 2008).

DSS and BI systems are "popular" in practice and academia and subsequently there are "a lot of environments" that relate to DSS/BI. This thesis however focuses on BI as an operational tool with focus on (business) environments where localised decision making may be of advantage as opposed to a more centralised approach.

BI is a central element of this research and the "starting point" for the research. Therefore it is important to have a definition of BI in context of this research. The next section reviews on definitions found in literature and describes commonly found components of BI systems. This is followed by an overview of key literature that describes development and trends and how those lead to the research gap.

2.2.2 Definitions

DSS and BI still lack a common definition and the constant change and progress in the DSS field make it difficult to find general definitions (e.g. Arnott and Pervan, 2008; O'Leary, 2008). Kimball & Ross (2002) and Moss & Atre (2003) suggest definitions of BI/DSS which are frequently cited:

> *"A generic term to describe leveraging the organization's internal and external information assets for making better business decisions." (Kimball and Ross, 2002, p. 393)*

"BI is neither a product nor a system. It is an architecture and a collection of integrated operational as well as decision-support applications and databases that provide the business community easy access to business data." (Moss and Atre, 2003, p. 4)

Both definitions have in common that they stress the holistic character of BI and that technical as well as non technical aspects are important in BI. Non-Technical aspects of BI are surely important, however these are not considered in this research. The focus is on technical issues of BI.

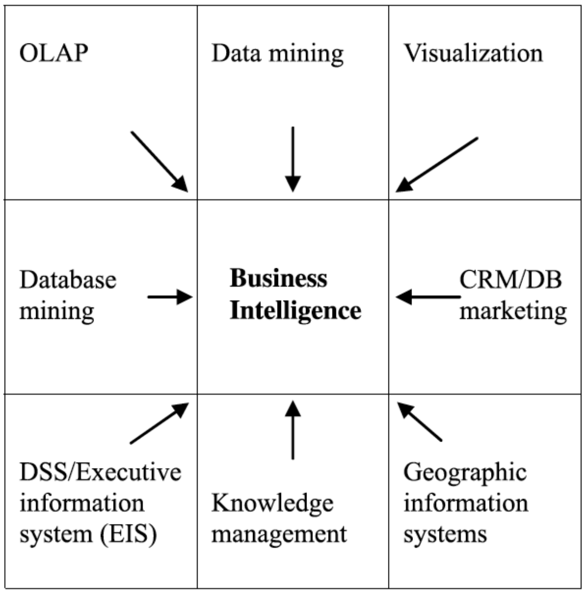

Figure 2.1 BI Components (Negash and Gray, 2008, p. 177)

Negash et al. (2008) argue that BI is the result of several innovations and each iteration resulted in more sophisticated concepts. Figure 2.1 summarises some of the key concepts like Executive Information System (EIS) or Geographic Information Systems (GIS) and technologies (e.g. OLAP and Data Mining) that relate to BI. Commonly found components of a BI system are:

Data Sources / Transactional System (OLTP)

Data Sources (e.g. transactional systems) technically do not belong to the BI system itself. However, internal and external data sources build the input for a BI system. Examples for source systems are Customer Relationship Management (CRM), Enterprise Resource Planning (ERP) or Point of Sale (POS) systems, but also external third party data (e.g. market research, government, etc.).

Extract Transform Load (ETL)

The ETL process is the connection between the operational sources and the DW. Tools that help to implement the ETL process support developers to connect to the DB (operational) systems to cleanse the data and transfer the data into the DW.

Data Warehouse (DW) and Data Marts

BI is a data driven form of a DSS and a central element is the Data Warehouse (DW). Technically a DW is a database (DB) however it is used differently to operational DBs. A DW is an integrated non-volatile data storage that may contain redundant data over long periods of time. Data from operational sources is cleansed, copied to the DW and stored in the DW. Usually a DW stores this data permanently to allow a historic and holistic view on an organisation. The data in the DW is the basis for various types of reporting and analytics supports prediction and forecasting activities.

Online Analytical Processing (OLAP)

OLAP describes the way to access the (multidimensional) data in a DW. It allows “slicing and dicing” and drill down into data. A decision maker can analyse data from different perspectives (e.g. sales by month by region) and arrange data in a way that helps best to solve a certain problem.

Reporting

Reporting is a basic feature of BI systems, but a popular and frequently used feature. It allows the design of pre-defined reports that can be accessed by employees (e.g. intranet). Reports are usually updated at fixed intervals (e.g. daily, weekly, quarterly, etc.). In addition to pre-defined reports, ad-hoc reporting functionality might be accessible for some users (e.g. Business Analysts).

Data Mining (Analytics/Forecasting)

Analytics means different things to different groups. In this relation other key words like: AI, Forecasting, Predictive Analysis etc. are used. For most implementations it is safe to say that analytics refers to methods and models that exceed basic descriptive statistics. Tan et al. (2005, p. 2) describe Data Mining (DM) as "… the process of automatically discovering useful information in large data repositories.". Data mining techniques can be applied to data in different areas (e.g. business, medicine, science). Examples of applications of DM in a business context are customer profiling or fraud detection. Generally DM techniques can be used for classification, association analysis, cluster analysis, regression or anomaly detection (e.g. Tan, et al., 2005; Vercellis, 2009). Mining techniques that are often found in BI systems are for example, Artificial Neural Networks, Time Series or Bayesian Networks algorithms. Microsoft SQL Server Analysis Services (which is used for the prototype) for example, provides Time Series and Artificial Neural Networks algorithms.

2.2.3 BI Developments and Challenges

Business Intelligence (as a form of DSS) has to evolve with its environment, the people that use the system and the technology that is used to implement the system. The concept emerged in the 90s and since then business has changed, technology has improved and new (business) challenges have emerged. As technology improved, so did BI and new and improved concepts have been developed over recent years to better address business needs. Yet, there are still opportunities to better support organisations and their decision makers.

DSS and BI systems are traditionally concepts that are used to support strategic and tactical decisions but did not support decisions at an operational level (There are specialised systems, like Real Time DSS, that aim to support more operational decisions). Businesses today constantly generate more data. This is partially related to new technologies that make data generation and storage possible and/or easier (e.g. barcodes, RFID, Internet / eCommerce, loyalty cards) and regulatory requirements that may require certain types of reporting and long term data storage (e.g. Accounting, Tax, Contracts etc.).

Edmunds and Morris (2000)[1] present a review of literature about the problem of information overload in business organisations. They found that data is often available in abundance, yet it is difficult to obtain useful and relevant information. Barone et al. (2010) similarly argues that data is readably available but "meaningful and productive" use of data can be difficult and still require "significant efforts" of IT staff.

The increase in available data, which has a rather operational character (high granularity, frequent updates), and the increasing number of decisions create a complex environment in which a decision makers might suffer "cognitive overload" (Vahidov and Kersten, 2004). The issue of information overload of decision makers and general time pressure in decision making was also raised by Phillips-Wren and Jain (2007). In this relation Cassaigne and Lorimier (2006, p. 401) state that decisions become increasingly complex and that "Straightforward cause-effect relationships are now less easily found...". They want to stress the point that data, information and influence factors to be considered in a decision process have increased and make the decision more complex.

[1] Despite being published over 10 years ago, the article is on the Jan/May 2011 "SciVerse Top 25" article download list of the International Journal of Information Management - http://top25.sciencedirect.com/subject/business-management-and-accounting/4/journal/international-journal-of-information-management/02684012/archive/31/

Recently Sargut & McGrath (2011) discuss the issue of complexity in business in more depth. They argue that businesses today are not just complicated but complex and that this complexity can lead to three i.e. managerial challenges forecasting the future, mitigating risk and making trade-offs. According to the authors it is difficult if not impossible to make decisions in such environments and they write: "We are further hampered by cognitive limits to our understanding of the effects of other people's actions and our own. Most executives believe they can take in and make sense of more information than research suggests they actually can. As a result, they often act prematurely, making major decisions without fully comprehending the likely consequences for the system." (Sargut and McGrath, 2011, p. 72). Analytics tools that should support us "… haven't kept up." (Sargut and McGrath, 2011, p. 70).

Considering this "demand" in decision support it is not surprising that there is a trend towards the application of BI on a more operational level (Kemper and Baars, 2009). Terms like Real Time BI, operational BI, localised BI, embedded BI and similar have emerged over the last years (Azvine, Cui, Majeed, and Spott, 2007; Marjanovic, 2007; Michalewicz, et al., 2007). Despite the different terminology the general goal is to better support decision makers by acquiring data closer to the event (e.g. in process) and deliver more up to date information to the user. As for this research, all these concepts are collectively understood as operational BI.

Increasing competition, more agile economies and rapid changes in technology are two main factors that will drive Real-Time BI (RTBI) according to Azvine et al. (2005). It is pointed out that real time can have different meanings. Real Time can refer to "Zero Latency" processes, that data/information are current whenever they are accessed or that KPIs reflect the current situation. BI and RTBI have the same functionality; however RTBI can use "zero-latency" data and is capable of influencing business processes in real-time. In Azvine et al. (2006) the need and concept of RTBI is further explored. The authors stress the point that timely reaction to changes in the business environment is crucial, and that BI requires real-time access to data and should be able to adjust the business process in real time.

The RTBI concept is applied to Operational Risk management in Azvine, et al. (2007). Marjanovic (2007) argues that today's operational BI implementations are limited to some core business processes. To support employees that interact with customers, operational BI systems have to be better integrated with overall business processes.

This shift in focus of BI, from strategic towards operational decision support, causes a conceptual misfit between BI and the business environment, because those systems (operational and BI) are not as integrated as possible. Current technology would allow a higher degree of integration. Common practice is to use ETL processes to gather data from operational sources, which often means transferring data from stores or branches (e.g. retailer industry). These processes are usually scheduled at fixed times (e.g. after close of business, at midnight, weekend etc.) (Meredith, et al., 2008). This means that the DW is always slightly out-of-date. Meredith et al. (2008) report that attempts to continuously update DWs had limited success.

Rigby and Vishwanath (2006) discuss in particular changes in the consumer market. They point out that in the past successful companies followed a standardisation strategy to gain efficiencies. However this is changing towards a strategy of localisation, to better address local market characteristics. Sargut et al. (2011, p. 73) agree and write "In business, the problem shows up when companies try to predict customer behavior on the basis of average responses. On average, people loved New Coke, but the product ultimately flopped. It shows up when they fail to consider that outliers are often more interesting than the average case.".

Traditional BI with its centralised focus does not sufficiently reflect this operational and localised focus. This research aims to address this issue by using agent and multi agent technology to extend BI so that the concept better applies to challenges outlined above.

2.2.4 Summary (Business Intelligence)

Despite the difficulties of accurately defining or describing DSS and BI, there is agreement that BI is a valuable concept. Business and technology has advanced and BI has to adjust to those changes to support organisations and decision makers to leverage the increasing amount of data that is available.
The reviewed literature shows that the trend is towards a more operational and timely/flexible architecture of BI to better adjust to changing business requirements. This means that BI has to handle more data in less time; BI should be close or embedded into the business process and be able to execute decisions.

2.3 Agent and Multi Agent Systems

2.3.1 Introduction

The idea of small autonomous software units that execute some work for their user is not new. More than a decade ago these ideas emerged (Hess, Rees, and Rakes, 2008). This turned into a new software engineering field, often referred to as Agent Oriented Software Engineering (AOSE). The Agent, as a design metaphor, is core to this field (Poutakidis, Winikoff, Padgham, and Zhang, 2009).

Imam and Kodratoff (1997, p. 1) noted "… if a researcher or any curious person wanted to learn about intelligent agents, he/she might get confused after reading even a few papers …". Today 14 years ago this statement remains mostly true, despite considerable research efforts, agent oriented software development and engineering presents itself as a fragmented research field with a lack of agreement on definitions, standards and best practices in key areas (DeLoach, 2009).

For example DeLoach (2009, p. 380) writes about "... the lack of a common set of notations and models, and the lack of flexible, industrial strength methods and techniques for developing multi-agents systems.". In the same issue Winikoff (2009,

p. 403) states "..., we are now in the position where there are a number of methodologies that are well developed."
This example shows how divergent opinions are. However there is some agreement to address this and advance agent technology both in academia and industry. Some 'marketing' actions are suggested for example by DeLoach (2009), Georgeff (2009) and Weyns et al. (2008). These actions include better communicating the advantages of AOSE over other methodologies and concepts used in mainstream software development, for example by publishing case studies to show how agent technology is applied to common (business) problems.

One way to use the agent metaphor is as a software architecture to design distributed systems. Lockemann (2006) claims that the Agent technology is "superior" compared to other technologies in highly complex business environments. Georgeff (2009) also argues that agents have an advantage, in regards to distributed architecture, in complex business situations. It is particularly referred to Service Oriented Architecture (SOA) and the shortcomings of the technology. SOA has some similarities with agents as a distributed design approach. However agents and AOSE is not just limited to distributed software design. Georgeff (2009), DeLoach (2009) and Weyns et al (2008) argue that agent technology is not sufficiently reflected in mainstream systems design and software development.

DeLoach (2009) stresses the point that the agent community should focus on the benefits of agent technology to develop complex, distributed and adaptive systems instead of overly focusing on agent definitions. This is part of the motivation of this research.

This section will briefly describe how agent and multi agent systems are understood in the context of this research. This will eventually help to establish why agent technology is a means to address the research questions. The definition is followed by an overview of the Microsoft Axum programming language. Further research on debugging and evaluation options of agent systems is presented.

2.3.2 Agents

Padgham and Winikoff (2005, p. 4) write that "It is important to realize that, like other software technologies such as objects, agents are not magic." Wooldridge (2001, p. 15) puts it into context and makes it more applicable in a technology environment:

> *"An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives" (Wooldridge, 2001, p. 15)*

This definition is still broad and abstract, however it explicitly says that an agent is some form of an artificially designed computer system (software or hardware) and that it has an objective. A second, less ambitious definition by (Russell, et al., 2002)

> *"An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators"*

Figure 2.2 illustrates this definition, having one agent interacting with its environment.

Figure 2.2 – Agent (Lockemann, 2006, p. 21)

The definitions listed are very abstract and can have very different meanings in different contexts. This is not very surprising as one core promise of agent technology is its adaptability and versatility. Yet it is difficult to use such broad definitions to work with.

Another approach to define (or describe) agents is to list the features or attributes of one. Symeonidis and Mitkas (2005) for example list nine characteristics that an agent may have.

Characteristic	Description
Autonomy	The agent can act to some degree autonomously and does not require (constant) supervision or external control
Interactivity (Reactivity / Pro-Activeness)	How the agent interacts with its environment. **Reactivity**: agent observes environment and reacts to it **Pro-**

	activeness: agent actively tries to reach a goal (depending on design objectives)
Adaptability	Sense other agents in environment and adapt to the environment
Sociability	Communication and relation with other agents (e.g. companionship, friendship and affability)
Cooperativity	Working towards the same goal with other agents
Competitiveness	Competing with other agents (e.g. auctions)
Temporal continuity	Robust design to ensure availability of agent
Mobility	Ability of agent to move between environments and to preserve its state
Learning	Ability to learn from past experiences and better react in the future.

Table 2.1 - Agent Attributes (Symeonidis and Mitkas, 2005, pp. 42-43)

Similarly Padgham and Winikoff (2005) list seven attributes (in bold) to define an agent. According to their definition an agent must be **situated** in an environment and act **autonomously** (i.e. not controlled externally). The agent must be **reactive** and respond to its environment but also show initiative (i.e. being **proactive**). It must be **flexible** in choosing actions to pursue its goals. To be able to "survive" an agent should be properly designed and **robust** (e.g. recover from failure) and capable of **social** interaction (with other agents). These characteristics are seen as central to agents. The authors mention other characteristics that might be important to some agents but not all, "less central" characteristics. They use rationality as an example for a "less central" attribute.

The importance and value of certain attributes or characteristics can be quite different. To give an example, Symeonidis and Mitkas (2005) list learning as a core agent attribute, however Padgham and Winikoff (2005) write that learning can be "disastrous" for some applications without giving an example. In the context of the proposed MAEBI system learning it is expected that learning does play a major role. With focus on the proof of concept implementation, "learning" (in form of an ANN) is used to determine the product prices.

Hess et al. (2008) mention the divergent definitions of agents in literature and provide a description (definition) of agent systems that is "useful" in the context of DSS. Because agents can be useful in many different contexts the authors argue that a reference point is required to define agents. This reference point must describe the represented object (e.g. person / user), the task and the domain the agent has to "work" for. Further they distinguish between essential agent characteristics and those who characteristics "empower". Essential characteristics are goal orientation, persistence and reactivity. For an agent to be useful it must have a goal (or goals) and try to reach this goal state and then maintain it. To be able to maintain that state, the agent must continue to exist (persistence) to do so. Lastly, agents can face different situations in their environments and have to be able to react to those changes. Mobility, Intelligence and Interaction / Communication are empowering characteristics and can allow the design of more versatile agents. Agents that have the ability to move to remote sites (i.e. environment) may help to better utilise existing (hardware) resources. Intelligence is a characteristic that is often mentioned in relation with agents and may help the agent to pursue its goal more efficiently with less supervision (e.g. user or designer/developer). The ability to communicate and interact with other agents can highly desirable for example in instances where a task is "too big" for one agent and that agent can delegate parts to other agents.

A third way to define agents is used by Knapik (1997) by grouping agents on their usage or function. It is questionable if this approach is very practical with increased usage of agent technology in different fields. It is likely that this would lead to numerous different agent/MAS groups and problems to classify a specific agent/MAS.

2.3.3 Multi Agent Systems (MAS)

The benefit of agent systems usually stems from the fact that multiple agents work in parallel or co-operate to achieve some desired outcome, which an individual agent could not achieve. The term Multi Agent System seems to be used rather ambiguously in many situations, as it is not always clear whether a system of agents is meant or a software runtime environment.

Symeonidis & Mitkas (2005) note that it depends on the actual application / context whether or not a (single) agent solution or a multi agent solution is required/suitable. An MAS is a network of agents, which may collaborate but are autonomous in their decision making. It is argued that a single agent cannot efficiently manage large amounts of data and is limited in intelligent behaviour.

Timm et al. (2006) list communication, interaction, structures and roles as key aspects of a MAS. MAS runtimes usually provide services to agents like communication and enable message passing, thus enabling or facilitating social aspects of the system.

A 'Blackboard' is a known means of communication in software systems and also used in MAS. It presents a simple and flexible solution to share knowledge across the agent system (between the agents). A blackboard can be implemented using different technologies (e.g. database). It allows agents to post and receive messages to and from the board (Figure 2.3). An advantage of this approach is that it is usually easy to implement with existing technologies (e.g. relational database system). Each agent needs an ID (e.g. GUID[2]) and it has to be able to write and read from the database. Direct communication (like P2P) between agents is not required.

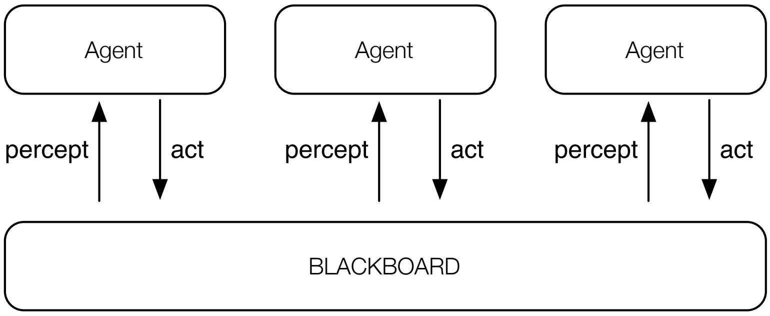

Figure 2.3 blackboard communication - adapted from Timm et al. (2006, p. 39)

The blackboard approach will be used to enable communication in the pMAEBI system in chapter 6.

2.3.4 Development Tools / Agents Platforms

Georgeff (2009) warns that the agent community should not focus too much on new languages and rather extend old languages. However, agents and agent oriented development brings requirements that are not necessarily addressed by 'conventional' programming languages. Also, some developers might perceive it as strange if there is no fit between model and code. For example one has to initiate a (OOP) *class* in code, which represents an *agent* in the model. Some agent requirements or constructs might be hardly or not at all achievable in standard languages. One option to facilitate agent based development is to use software frameworks in object oriented laguages. Frameworks can help developers by simplifying certain agent related programming tasks. Gustafsson (2009b) argues that such frameworks (extension of current languages) do enable developers to use agent

2 Globally unique identifier

concepts in OOP languages, however these frameworks do not force developers to adapt to the new programming metaphor. A language that is built around the agent metaphor would force developers into the approach.

To facilitate the development of multi agent systems, infrastructure support is required that provides necessary services to the agents. A runtime or individual services can be custom built, however using a pre-existing system should simplify the development process and enhance interoperability. Object Oriented Programming (OOP) is the quasi standard for modern software development. OOP - Programming languages like C++, C#, Java are ubiquitous, Tools (IDEs, Debugger, etc.) and methodologies like UML are well-established and available, but agent tools are very limited. Braubach et el. (2006) present a systematic review of agent oriented development tools and noted that the majority of agent tools are runtimes or libraries. However close to 50% of those were under current[3] development.

Sudeikat et al. (2005) suggest a framework of platform dependent and independent criteria to compare the growing number of methodologies to develop agent-based systems. They further write "… the differences between available implementations are too fundamental to be ignored" (Sudeikat, et al., 2005, p. 126). Pokahr and Braubach (2009) analyse current agent based development tools and conclude that the number of development tools is low compared to OOP tools. Also, the agent tools available are very specific, which means that they usually only support one agent methodology.

The Jade MAS platform is one of the most commonly known agent platforms. Telecom Italia originally initiated the development of Jade with the aim to validate early FIPA[4] specifications. Following further support by the European Commission, the Jade team was able to continue their work to develop a full FIPA compliant

[3]No publication with the system in the 2 years before publication of their paper in 2006 (Braubach, et al., 2006).

[4] Foundation for Intelligent Physical Agents

agent platform. Today, Jade is an open source system, maintained by the Jade Governing Board (Non Profit Organization) led by Telecom Italia and Motorola (Bellifemine, Caire, and Greenwood, 2007). JADEX was and is developed at University of Hamburg, primarily by A. Pokahr and L. Braubach (2005). It is based on JADE as platform, and adds a Belief Desire Intention (BDI) Reasoning Engine.

2.3.4.1 AXUM

AXUM (Gustafsson, 2009a) is a .Net based Agent oriented programming language developed by Microsoft Research. Like most other agent frameworks here, AXUM is a research project; however it is implemented as part of the Microsoft .Net Framework. As Weyns et al. (2008) point out, although the agent community proposed and partially implemented methodologies and frameworks, there is a "Lack of Integration with General-Purpose Technologies,..." (Weyns, et al., 2008, p. 5). Axum in combination with the .Net Framework / CLR Runtime and Visual Studio as IDE does address this.

Axum is a prototype programming language developed by Microsoft (Gustafsson, 2009a, 2009b). The language follows the actor model and is conceptually influenced by languages like Scala and Erlang. Syntactically however Axum is very similar to C#, integrated in the .Net Framework and implemented on top of the Concurrency and Coordination Runtime (CCR). Despite similarities to C#, Axum is not an object oriented programming language. OOP constructs, like Classes, Interfaces and Structs do not exist in Axum. Instead the language uses Agents, Domains and Channels as building blocks (Gustafsson, 2009a).

One aim of the developers of Axum was to provide a language that forces software developers to follow parallel and concurrent design instead of providing libraries that just enables developers to do so (Gustafsson, 2009b).
Axum defines four major components that are different to OOP, namely Agent, Channel and Ports, Domain and Schema (Gustafsson, 2009a).

In Axum an agent is the organization unit that implements a channel. The agent is the construct that executes code and can, depending on the access rights, change the state of the domain it belongs to. Agents can have Read/Write, Read or No access on the domain state.

Channels allow message passing in Axum. In OOP programming a channel is roughly comparable with an interface of a class. Note that a channel is instantiated rather than the agent directly. Each channel defines one or more ports. A port is an input (into the agent) or output (return value from the agent) of a specific data type.
A domain is an isolation unit that separates the memory of concurrent application parts. Each Axum application has at least one domain that acts as the 'start' for the application. Agents and Object (e.g. Variables) are defined in a domain.

Schemas in Axum are used to define sets of data that can be transferred between agents. The syntax of a schema is similar to a channel, however the purpose of a channel is to allow communication between two agents and a schema is to describe a data "container". In contrast to 'normal' variables or data types, schemas provide basic rule support, for example fields can be defined as required or not empty.

Axum is a very interesting approach to agent oriented development. It integrates with tools that developers already use and the syntax is very similar to mainstream languages. This means that developers can focus on the new agent metaphor without spending too much time and effort to learn about new tools. The integration with the .Net framework was limited in the Axum preview. In general, it

will be challenging to use libraries in agent systems, as the functions offered have to be atomic in nature.
Microsoft has already announced that they will not continue to pursue a production release of Axum. However, the fact that a ‘big player’ like Microsoft has shown interest in the field of agent oriented software development might give the agent community some momentum.

Chapter 5 describes the implementation of the prototype and gives some additional information about Axum.

2.3.5 Debugging, Testing & Evaluation of Agent Systems

> *“As soon as we started programming, we found to our surprise that it wasn’t as easy to get programs right as we had thought. Debugging had to be discovered. I can remember the exact instant when I realized that a large part of my life from then on was going to be spent in finding mistakes in my own programs.” - Maurice Wilkes*

In every development process debugging and testing are crucial steps to ensure that the program executes according to its design expectations. In most cases, but certainly in the academic environment, some form of evaluation should be used to determine feasibility and performance of the software artefact.

Debugging, testing and evaluation of software is generally difficult. In case of multi agent systems these tasks become even more challenging, as MAS are inherently complex. Experiences, best practices and tools are not yet established, not yet agreed on or not yet available (Hanks, Pollack, and Cohen, 1993; Poutakidis, et al., 2009; Timm, Scholz, and Fürstenau, 2006).

The aim of testing and debugging is to identify differences in the actual execution of the software to the expected behaviour. This process is difficult in multi agent systems because of the modular architecture and the autonomy of the agent(s) (Poutakidis, et al., 2009). They present two tools, one for debugging and one for testing. The underlying framework should be applicable to a range of agent systems and methodologies, however the artefacts (i.e. the actual code implementation) are mostly system dependent and have to be implemented to the specifics of the system under interest.

Zöller et al. (2006) discuss benchmarking of multi agent systems as a special form of evaluation. They focus on comparative evaluation but suggest descriptive evaluation for systems that are new and where a 'baseline' system does not exist. Other aspects of evaluations are Time, Method and Focus. Time refers to when the evaluation took place, during the development (testing individual components of the system) or after the development (evaluation of the complete system i.e. all components). Methods are distinguished between laboratory and real word benchmarking. In particular if real world benchmarking is too risky or too costly, laboratory benchmarking has advantages. Simulation is mentioned as an important technique in this context. Whether part or the whole system is tested defines the Focus of the benchmarking analysis.

Similarly, Hanks et al (1993) argued for test beds and controlled experiments to test multi agent systems. Performing real world evaluation on agent systems that is by either using real data or implementing the solution in a real word setting (i.e. with a business partner) is often not plausible or possible. In the context of this research those business partners would be retailers. Retailers do not necessarily share all the required data and even if they would, the problem would be the significant size of the datasets and the computing resources required to analyse and process the datasets.

Theodoropoulos et al. (2009, p. 77) argue "Multi-agent systems (MAS) are often extremely complex and it can be difficult to formally verify their properties. As a result, design and implementation remains largely experimental, and experimental approaches are likely to remain important for the foreseeable future. Simulation is therefore the only viable method to rigorously study their properties and analyze their emergent behavior." Helleboogh et al. (2009) quote Himmelspach, Rohl and Uhrmacher (2003) and state that simulation is a "… safe and cost-effective way for studying, evaluating and configuring …" of multi agent systems (Helleboogh, et al., 2009, p. 2)

Helleboogh et al. (2009) continue and focus on software-in-the-loop simulation for dynamic environments. The requirement for this type of simulation is that the environment is constantly changing (i.e. dynamic) and that the multi agent system is embedded in the simulation (in the loop). Adapted to this research, one part of the testbed is the MAS system (the system to be tested) and the second component is the simulation that acts as the retail environment. The simulation not only generates data (sales data) and feeds those into the MAS system, but the MAS system processes that data and can feed data (changed sales price) back into the simulation system. Both components have interfaces that connect them to databases as they would in the "real word". The "Real World" box in Figure 2.4 should indicate that, conceptually, it should be possible to take the MAS system and place it in a real world setting.

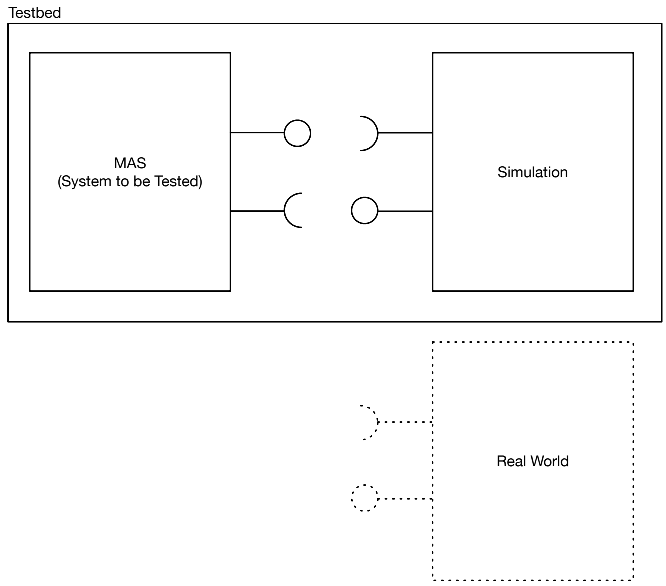

Figure 2.4 Software in the Loop Testbed

The advantage of simulations in general, compared to other methods, is the possibility to analyse the system in its context. It further allows defining test cases and scenarios, which might be difficult to find in real data.

Baydar (2003) for example describes a simulation based approach to optimize store performance. Neri (2007) for example used agent based simulation to analyse customer behaviour under information diffusion. Zenobia et al. (2009) review literature about the emerging field of artificial markets as a form of agent based social simulation. They conclude that the approach is an interesting option to explore market dynamics. However they do mention that such models can be complex and might be too sensitive to certain start up parameters.

Another example for a simulation based study is described by Zimmermann et al. (2006), They present a system to identify disruptive events in supply chains using agent technology to improve reaction times and improve supply chain performance. They designed a prototype and used a custom simulation to test their system.

The approach chosen for the evaluation of the pMAEBI system is an experimental/simulation approach (Herrler and Kluegl, 2006; Vitolo and Coulston, 2004). Output of the simulation system is demand data (Market Basket Data), which in turn is the input of the pMAEBI system. This is the type of data that is collected at retail checkouts. Literature refers to that type of data with different terms, like sales data, (retail) market basket data or scanner data, which is the type of data that is the main pillar of a majority of research conducted in this area.

To test the application we used a software-in-the-loop simulation approach. This means that the simulated Customer is part of the pMAEBI application. Simulation of customer demand is challenging for two reasons. Firstly, marketing models are generally qualitative and subsequently not transferable or only partially transferable to actual simulation code and secondly there is no general model that describes the buying and decision making of a customer. This generally results in simplified models.

2.3.6 Agents and DSS/BI

The decision process itself has become steadily more complex (e.g. Cassaigne and Lorimier, 2006). This means in respect to the analysis and data mining techniques employed in BI that one method is rarely sufficient to achieve a satisfying result. Systems that combine different mining methods are referred to as Hybrid Intelligent Systems (Z. Zhang and Zhang, 2004). They further argue that the design of such systems is complex as they consist of a large number of different components. Agent and in particular multi agent systems present a design paradigm that has the required functionality and characteristics to design such hybrid systems and can improve BI in terms of its functionality and the other mentioned problem areas.

Rabelo and Klen (2002) address in particular the problem of BI in the context of Supply Chain Management (SCM). They developed the SC^2 system that supports the communication between suppliers and the business. The authors see BI more as a data source.

Lavbic et al (Lavbic, Rupnik, Bajec, and Krisper, 2007) propose to use agents as a DSS backend. The focus is on data integration and information retrieval and the use of anthologies and semantic web technologies. The authors mention a case study of a mobile phone operator in Slovenia, but do not provide much detail about the case and how the system is realised and implemented. Also they do not report about testing or performance evaluation.

Cao and colleagues introduce the concept of Agent Mining interaction in Zhang et al. (2005) and Cao et al. (2007) followed by a more in depth book (Longbing Cao, 2009). They suggest that agents can be useful in data mining as some requirements of DM technology and agent technology overlap or enhance each other.

2.3.7 Summary (Agents)

This section presented an overview of agent and multi agent technology as understood in the context of this research. Agent technology is a very broad and interdisciplinary research area, with significant potential.

There is no broad acceptance of a general agent definition. It rather seems that the definition will vary in different contexts or just depends on the perspective of the beholder. Yet, based on literature a working definition for Agent and Multi Agent systems was established (for this reseach) and required and optional characteristics of an agent were defined.

As DeLoach (2009) pointed out, for every agent implementation there is an OOP way to do it. This research should contribute to the body of research and add to the experiences of multi agent systems development and help to better describe and synthesise the advantages of Agent Oriented Software Engineering (AOSE) over Object Oriented Programming (OOP).

Axum, a prototype Agent oriented programming language developed by Microsoft, was presented in this section. The Axum language was used to implement the prototype. Axum adopts enough agent concepts to be considered an agent language, yet keeps many of the known components, like IDE, Syntax, Runtime and a – for an agent language – considerable framework. Using MS technologies it would be executable on most Microsoft platforms.
Considering current IT developments, like SOA, Cloud, Multi Core CPUs/GPUs on the one hand and increasing business pressure on the other hand, software systems have to handle complex business processes and adapt to changes within an organisation and to the environment of that organisation. Core to agent technology or agent oriented software development (AOSE) is this flexibility and versatility to adjust to changes in its environment. As such the technology has the potential to allow the design of flexible business software, here in particular to design a flexible and dynamic BI system that can be embedded into individual (local) problem domains rather than following a centralised approach.

2.4 Retailing and Product Pricing

2.4.1 Introduction

One of the driving forces to improve Business Intelligence and expand the reach of BI implementations is the complexity in business or business processes. Pricing is one, particularly complex and vital business challenge. This section will give a brief introduction into retail pricing, the problems involved and how recent developments increase the importance of technology in pricing.

2.4.2 Retailing

Krafft and Mantrala (2006) describe retailing as an exciting, complex and vital industry in most countries, developed and emerging. They list factors like customer change, competition and technology as drivers for significant change within the industry. Fisher et al. (2000) reported on the early developments how retailers adopt technology to optimise their operational business. The authors referring to this (at that time) new development to "rocket science retailing". Today technologies like barcodes, ecommerce and RFID, to name a few, had a significant impact on retailing. The pressure on retail stores and chains however is not easing. Simon et al. (2005) for example report that the average profit margin of a European retailer is just 0.7%.

Retailers continue to increase their investment in technology to remain competitive. Retailers have three different options to increase profits, (1) lower production costs, (2) increase market share or (3) adjust price (Dolgui and Proth, 2010). Retailers traditionally focus on reducing cost and increasing market share (Dolgui and Proth, 2010; von der Gathen, et al., 2005). McIntyre and Miller (1999) see stocking and pricing decisions as one of the most central problems of retailing. Despite this importance, price is not well understood (Grewal, Levy, and Kumar, 2009). Marketing classically orients itself along the 4ps (Product, Placement, Promotion and Price). Of those Price is 'special' as it 1) connects supply and demand (Figure 2.5) 2) is the only 'P' that actually makes money 3) Price is the easiest (sometimes instantly) to adjust (Rao, 1984).

Figure 2.5 illustrates how the price of a product links supply and demand (and vice versa) side of a market.

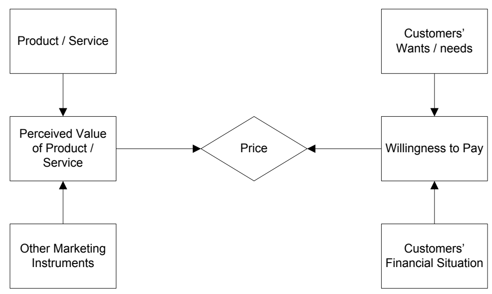

Figure 2.5 - Price Value Context Source: (H. Simon, 1989)

A frequently cited study by Marn and Rosiello (1992) showed the impact of price on company's profits. They showed that a mere increase of 1% in price results in an average of 11% in additional profit. This study is the rationale to focus on pricing. Applying these findings to retailers, the gain is usually higher, and profit increases up to 70% are possible (von der Gathen, et al., 2005)

2.4.3 Pricing Strategies

There are many different approaches and strategies to set prices. The roots of pricing theory go back to the economic supply and demand theory, which was developed in the 19th century, when products were commodities and purely fulfilled customers' basic needs.

Today however these economic theories might only apply in small markets, e.g. where professional buyers trade commodities. In most other markets price, supply and demand are much more complex to determine as consumers do not just buy because they need, but because they "want to" (Mercer, 1996). Simon et al. (2005) are of the same opinion and use the term "classic pricing theory" and stress the microeconomic objectives of these theories and their limited applicability.

Philips (2005) lists three commonly used pricing approaches of retailers and key focus of those approaches (Table 2.2). "Cost-plus" is an approach that focuses on the actual cost of an item to the retailer and adds a predefined margin (e.g. $1 cost + 15% profit = $1.15 retail price). This approach is likely to miss out on profit opportunities as competition and the consumer (what the consumer thinks the product is worth) are ignored. The approach is usually very simple to implement, however depending on accounting practices, it might be difficult to identify the direct costs of a product.

Approach	Based on	Ignores	Liked by
Cost – plus	Costs	Competition, Customer	Finance
Market based	Competition	Cost, Customers	Sales
Value based	Customers	Cost, Competition	Marketing

Table 2.2 - Alternative Approaches to Pricing (Phillips, 2005, p. 22)

A market based approach uses the competition as reference point for pricing, which makes it favourable for sales people as they can sell products by a "its cheaper here" strategy. However the approach ignores costs and the customer. The approach can lead to price wars when all sellers try to be cheaper. Market based pricing focuses on the customer and what he/she is willing to pay, which is ultimately the factor that matter the most.

In addition to those pricing approaches, retailers follow 'general pricing strategies' that aim to communicate a 'price image'. "Everyday Low Prices" (EDLP) and "Hi / Low" (HiLo) are the most common ones. EDLP describes a strategy where retailers offer their products constantly at a low price; HiLo describes a pricing strategy where prices are kept at a rather high level and lowered temporarily (discounts). The distinction of pricing strategies and approach becomes difficult as retail formats and concepts, and with it the pricing strategies 'blur' (Fox and Sethuraman, 2006; M. Levy, Grewal, Kopalle, and Hess, 2004). It is not surprising that pricing was rated by managers as the area with the highest problem pressure (H. Simon and Dolan, 1996).

Despite this importance and impact of pricing retailers still follow, often simple, rule based pricing approaches that frequently generalise pricing decisions. Levy et al. (2004) mention this "system wide character" of pricing decisions. This "strategy" is surprising, as already mentioned, price is one of three profit drivers, together with volume and costs. Retailers are more focused on reducing costs and increasing volume than improving pricing, which shows a greater impact on profits than the other two drivers (von der Gathen, et al., 2005).

Levy et al. (1998, p. 82) describe the pricing process for retailers as a "nontrivial task". Often pricing is not addressed appropriately, which results in a "cost plus something" approach or other rule of thumb strategies.

Schwind (2007) (Figure 2.6), classifies pricing into two main categories, static and dynamic. Static refers to pricing that does not change over time (or price changes in the long run) and the buyer has no influence on the pricing process. Dynamic on the other hand refers to approaches which include the buyer directly (interactive pricing) or indirectly (dynamic price posting). Indirectly in this context usually means that the seller analyses sales/demand data to adjust prices accordingly.

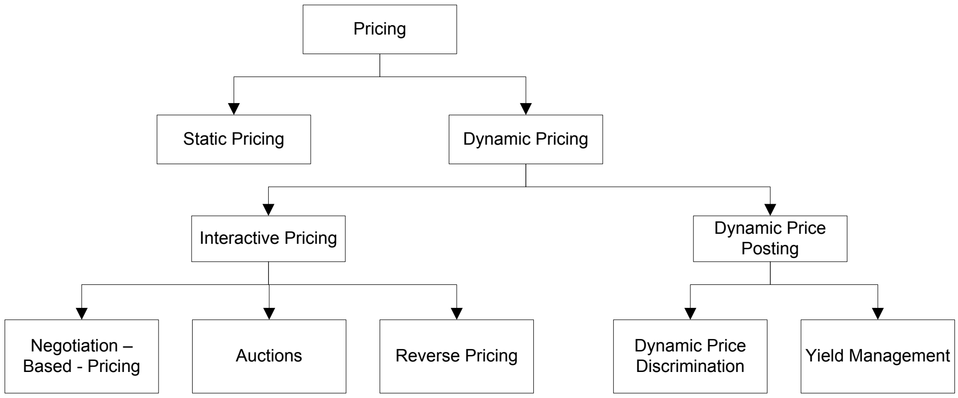

Figure 2.6 - Taxonomy of pricing mechanisms Source: (Schwind, 2007, p. 28)

Elmaghraby and Keskinocak (2003) state that "dynamic pricing posting" is current retail pricing practice. The trend towards a more dynamic and customer focused pricing, is driven by (1) advances in IT, in particular scanner based checkouts and (2) the advent of online stores (easy to change prices) which brought the dynamic pricing approach back into marketing focus (Elmaghraby and Keskinocak, 2003; Schwind, 2007).

In particular the emergence of the Internet and with it eCommerce and eBusiness triggered a new interest in pricing and marketing methods. Online stores, in comparison to 'brick-and-mortar' stores, allow the individual tracking of customers, adjustment to product offerings and price changes at no cost. Concepts and approaches that were not possible before now become, at least, feasible.

Technologies and concepts found their way from those virtual stores into real stores. Ravi et al. (2010) stress the point of increasing amounts of data available in the retail sector and the application of soft computing and real time systems in retailing. They also conclude that retailing has become more technology driven. Those developments make pricing even more complex for retailers. Dixit et al. (2007) present a taxonomy of IT enabled pricing strategies, strategies that were not possible or feasible before. However, having more data and computational capabilities available, these strategies present interesting approaches to increase profits.

Rigby and Vishwanath (2006) in this context describe how the retail environment is changing. They argue that consumers are becoming more diverse and that a 'one size fits all' approach is no longer suitable for retailers to stay competitive. This is a major shift for retailers that traditionally tried to standardise as many aspects of their operations to gain efficiency and pushed this 'thinking' through the supply chain.

2.4.4 Pricing Decision Support Sysems

To help retailers to make pricing decision for the many Stock Keeping Units (SKUs), pricing DSS systems are available. The idea for such systems appeared many years ago for example in Breath and Ives (1986). However those systems did not fully address retailers needs (Montgomery, 2005).
Trivedi (2011) argues that retailers gather data with high granularity, but by using aggregates in data analysis some of the captured information is lost. In this relation Phillips (2005, p. 33) states the "analyst with a spreadsheet" method of pricing decision making begins to break down. Similarly Davenport (2006, p. 102) writes about the "business reality" where employees use self developed Excel sheets that are often emailed around in different versions and present a "breeding ground for mistakes"

Baydar (2008) suggests the use of evolutionary computation to optimise store performance. It is argued that a more individual approach to give discounts to customers is more effective that using general store cards. A simulation is used to test the hypothesis and suggests that this approach can work.

Levy et al. (1998) describes the process of changing prices in the retail environment in all its complexity and challenges for managers and store personnel. Dasu and Tong (2010) investigate dynamic pricing policies and also note that it is difficult and expensive for brick and mortar stores to change prices. However new technologies like Electronic Shelf Labels (ESL) (e.g. Zentes, Morschett, and Schramm-Klein, 2007) that reduce the cost of re-pricing, as well as allow frequent price changes are not reflected in these studies.

2.4.5 Why Retail Pricing is a suitable area of application

To present, at least some, of the features and capabilities of the suggested MAEBI system, the concept has to be applied to a complex and distributed problem domain. Pricing in retail chains is certainly such a problem and it is highly relevant and applicable in a business sense.

To address the pricing problem systems are needed that combine 1) analytics 2) learning 3) adaptability and flexibility 4) localisation and 5) automation.

The MAEBI system provides 1) analytic capabilities with its Data Mining and Knowledge Discovery module 2) it provides learning capabilities 3) the agent architecture makes it flexible and aligned with business structure 4) to focus on local characteristics and 5) the decision execution module allows changing business processes, here pricing, in real time without human interaction.
MAEBI is likely to be applicable to different problem domains and pricing is surely one of those.

2.4.6 Summary (Pricing)

This section summarised some key aspects of pricing with challenges and developments for retailers. Information Technology is already a vital part of pricing and competitive pressure will continue to drive interest and adoption of technology to optimise business in general and pricing in particular. The inherently distributed structure of retail chains, the complexity and frequency and the number of pricing decisions to be made, presents an ideal test scenario to be used in an agent / business intelligence setting.

2.5 Chapter Summary

This chapter reviewed key literature in the fields of Business Intelligence, Agent and Multi Agent Systems, and Retail Pricing and presents the relevant knowledge base for this research. Each of these topics presents a current and highly interesting area in IS/IT research that is of particular relevance for businesses.

Business Intelligence is the term that describes the current state of decision support systems. The concept emerged in the early 90s and supports business to gather and process large quantities of data and turn this into information. However change in the business environment requires that such systems handle more data in less time to support more complex decision processes. Literature repeatedly argues that current BI systems do not address these demands appropriately.

Agent and Multi Agent Systems present a new technology with characteristics that allow the design and development of highly flexible software systems for use in complex environments. This technology is used in this research to design a new BI system that overcomes some of the issues that were identified in literature.

A brief overview of retail pricing was presented. Retail pricing is used as the domain to implement the prototype system. The price connects supply and demand and is a crucial decision for businesses. Small changes in pricing can have significant implications on profitability. In particular in industries with a high number of pricing decisions, like retail, IT systems can support by analysing large amounts of data. The number of decisions and the different influence factors make pricing a highly complex problem and should be able to show some of the aspects and advantages of the proposed system.

This research addresses the identified research gaps to help to develop BI so that the concept can leverage available data (and the investments made in technologies) and better support business and their decision makers in their work.

Chapter 3 Methodology

3.1 Introduction

To produce valuable research and to contribute to the body of knowledge in a certain discipline a researcher should define and follow a suitable research methodology. In recent years there has been some discussion about research in Information Systems about rigor and relevance and related to that whether design constitutes academic research (e.g. Galliers and Land, 1987; Nunamaker, Chen, and Purdin, 1990-91). Gallupe (2007, p. 1) for example wrote "Current IS research seems more concerned with 'how' the research is conducted rather than 'what' research is conducted and 'why'."

Arnott and Pervan (2008) focus on DSS research and reviewed DSS related literature. They note that DSS is a important research area in IS and a 'major' issue in IT practice; "… IT-based decision support can have a significant effect on the nature and performance of an organization." (Arnott and Pervan, 2008, p. 1). The authors suggest 8 key issues in the discipline. One issue is the relevance of DSS research and its applicability to the real world problems. It is argued that for the sake of rigor, relevance is often neglected. For example their analysis showed that 49.2% of the reviewed research had "low practical relevance or none at all."

It is not the purpose of this thesis, nor this chapter to discuss the underpinnings of research methodologies and philosophical justifications. Yet, this chapter should present a research methodology that is sound and based on literature as well as show rigor and relevance of this research. Because of that, this chapter explains and justifies the choice of research paradigm applied, the design of the methodology and methods and techniques.

Webster's Dictionary first definition of methodology is "the analysis of the principles of methods, rules, and postulates employed by a discipline". Jonker and Pennink (2009, p. 17) define a methodology somewhat more pragmatically as " … broadly speaking, the way in which a researcher conducts research."

This chapter will outline exactly how the research was conducted. First the research aims are re-visited to define and focus the goal and the expected outcomes of the research. Using Jonker and Pennink's (2009) Research Pyramid (Figure 3.1) as guiding framework, the paradigm and methodology is justified. Research methods and techniques are derived from Hevner et al. (2004) and Peffers et al. (2008).

3.2 Research Pyramid

To help to outline an appropriate research methodology, Jonker and Pennik (2009) suggest using the Research Pyramid (Figure 3.1) as a guiding tool. According to the authors the purpose of the pyramid can be described as "The key function of the pyramid is to help the researcher learn to consciously structure his approach to the research." (Jonker and Pennink, 2009, p. 25). It is argued that "…the researcher should be able to justify the reasons for this choice of a specific (research) approach and make sensible choices based on the different requirements of a particular question." (Jonker and Pennink, 2009, p. 22).

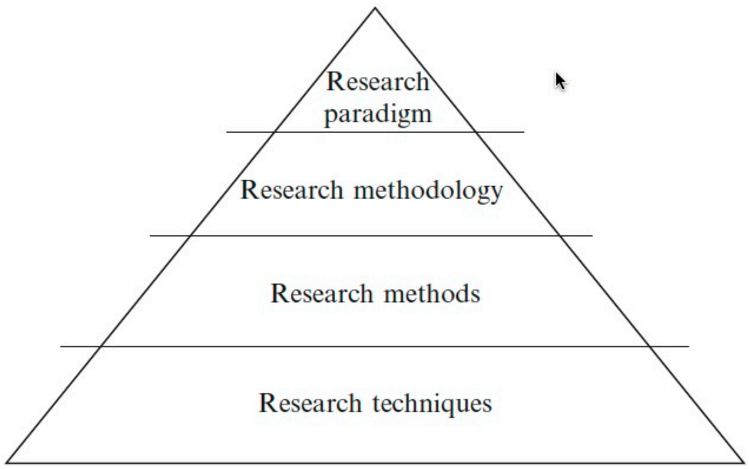

Figure 3.1 Research Pyramid (Jonker and Pennink, 2009, p. 23)

The research pyramid has four levels, Paradigm, Methodology, Methods and Techniques. It guides the researcher from rather abstract views to concrete application of techniques. At each level a researcher has to justify choices made.

Jonker & Pennink (2009, p. 21) nicely summarise the purpose of a research methodology as "The essence of methodology is structuring one's actions according to the nature of the question at hand and the desired answer one wishes to generate". Depending on the paradigm, a researcher has to 'map out' a methodology for the research at hand. This 'map' indicates the start point, the research problem, and how to get to the goal, the results. The methodology has to be further developed by choosing appropriate methods for the problem domain. Once a decision was made, the methods have to be implemented using suitable techniques.

3.2.1 Research Paradigm

What constitutes research and how 'reality' is perceived is different between professions. A research paradigm can be described as " . . . the underpinning values and rules that govern the thinking and behaviour of researchers." (Gummesson, 1999) in (Jonker and Pennink, 2009).

Vaishnavi and Kuechler (2007) quote Kuhn (1996) who argues that researchers in "'tightly' paradigmatic communities" may not be aware of the philosophical implication of the way they conduct research. Information Systems is a multi-paradigm discipline (Vaishnavi and Kuechler, 2007), which implied that the researcher should have some understanding of the chosen paradigm and the implications of this selection. This research follows the Design Science Paradigm (e.g. Hevner, et al., 2004; Nunamaker, et al., 1990-91). The goal of DSR is utility (Hevner, et al., 2004, p. 80) or progress (Vaishnavi and Kuechler, 2007) in a specific problem domain.

3.2.2 Research Methodology

Level 2 of the pyramid refers to the methodology, the way the research is conducted. It describes the steps of the research process and how to get from starting- to finishing point. The starting point is the problem and the finishing point is the result or insight.

The methodology for this research is developed based on Hevner's et al. (2004) Design Science Research Guidelines and incorporates Peffers's et al. (2008) Design Science Research Process.

3.2.3 Research Methods

A research methodology is a general framework and has to be filled with actual methods that describe 'how' a particular research is performed. The choice of methods should evolve from the choices made on levels above in the pyramid.

Hevner et al. (2004) suggest several research methods that are applicable to DSR in IS. Their suggestions are described in detail later in this chapter and the method of choice is justified. The method of particular interest here is experimentation / simulation and will be explored in detail later (section 3.4).

3.2.4 Research Techniques

Level 4 is the most concrete of the pyramid, and is concerned with the actual techniques (or instruments and tools) used to facilitate a method specified the level above. Jonker and Pennink describe techniques as "Techniques can be understood as concrete instructions for acting that have an explicit, compelling and prescribing character." (Jonker and Pennink, 2009, p. 34)

The 'instruments' and 'tools' used in this research is a self-developed test-bed simulation implementation that is in line with Hevner's suggested methods and MAS literature. Sections 3.4.4 and 3.4.5 will provide reasons for the choice; chapter 5 describes the implementation in detail.

3.3 Design Science Research (Research Paradigm)

Design Science as research paradigm in Information Systems emerged around 20 years ago in the early 1990s. Design as a research approach is strongly related to Simon's "Science of the artificial" (H. A. Simon, 1996). Using design as an approach of academic research is known in other disciplines, e.g. Architecture or Computer Science, where the approach is valued and proven (e.g. Hevner, et al., 2004; Nunamaker, et al., 1990-91). In the field of IS however, there are sceptics of the idea of design as a method of research (e.g. Galliers and Land, 1987).

It is to note that the purpose of this chapter is to define a research methodology that is based on established literature. It is out of the scope of this chapter to argue whether design and subsequently design science research constitutes research in IS.

However, there has been some change towards the application of design as a research paradigm in IS (Gregor, 2006; Hevner, et al., 2004; Offermann et al., 2009; Peffers, et al., 2008). March and Storey (2008, p. 726) write in the introduction to the MISQ special edition on design science "Design science research is increasingly recognized as an equal companion to behavioural science research in the information systems field.". Hevner et al. (2004, p. 79) suggested a conceptual framework to understand, execute and evaluate DSR in IS which is discussed below.

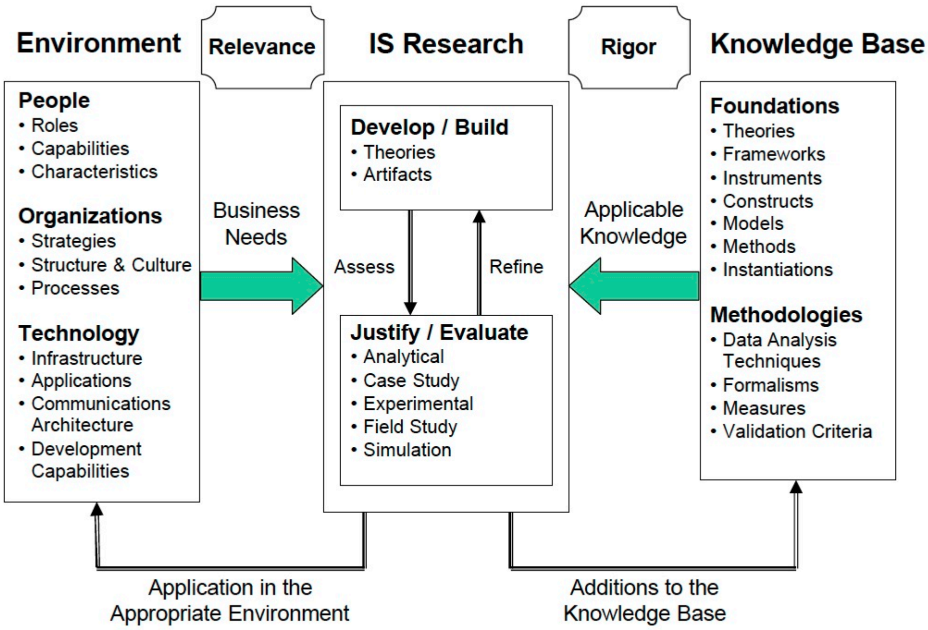

Figure 3.2 IS Research Framework (Hevner, et al., 2004, p. 80)

3.3.1 Business Needs

Hevner's Framework (Figure 3.2) illustrates the "Business Needs" as a major input or driver of IS research, emphasising the need for relevance of the research. This relevance can refer to People, Organisation or Technology. The main 'link' to the environment of this research is technology.

Computational Decision Support, in all its variations, is a crucial part of (business) software and IS research (Arnott and Pervan, 2008; Burstein and Holsapple, 2008). The research focuses on the BI (a data driven DSS) concept as an advanced set of tools to provide an architecture to deliver flexible decision making capabilities across organisations. Such technology is relevant for virtually all organisations in all industries; decision-making happens in every organisation.

However, as described in the literature review, BI is a complex set of technologies and processes that have to interact to work. The BI concept developed over the years and tools became readably available. Not just commercial offerings from Oracle, Microsoft or IBM (Cognos) are available, but different open source BI solutions like Pentaho or Jaspersoft have reached maturity and allow deployment in productive environments.

Decision Support affects or is affected by all three areas (People, Organisations and Technology) of the environment (left side of Figure 3.2). Without implying any judgement about importance and necessity, the impact on People and Organisations is not covered within the context of this research. This research focuses on the technological aspects of BI and Agent technology. If the research does show promising results further research has to be undertaken to investigate how such a system aligns with an organisation and its users,

3.3.2 Applicable Knowledge

Hevner et al. (2004, p. 80) argue "The knowledge base provides the raw materials from and through which IS research is accomplished." (Figure 3.2). The 'knowledge base' for this research is literature in the domains of DSS/BI, Agent and Multi Agent

Systems and Retail Pricing. Literature about DSS and particularly about BI is the 'starting point' of the research. The review of literature documented in chapter 2 summarises important (for this research) aspects of current research and issues in this context. Agent and Multi Agent technology are concepts that were identified as possible 'solutions' to advance BI and to address the identified issues. Lastly knowledge was drawn from the field of retail pricing, a complex and important business area, that is used to implement the proof of concept system.

3.4 Research Process (Research Methodology)

The second level of the Pyramid (Figure 3.1) is concerned with an appropriate methodology within the context of the research paradigm chosen previously. Jonker et al. (2009) quote Cobuild (1987) ". .. a system of methods and principles for doing something" as a definition for 'methodology'.

Peffers et al. (2008) suggest a DSR methodology that aims to be consistent with prior literature, provides a process model and a mental model for presenting the design science research. They further argue that such a process model provides support for researchers in DSR. They do not claim that this is the only way to do DSR, but that this is "a good way" (Peffers, et al., 2008). To justify that this is an appropriate choice for this research the process is evaluated against Hevner's DSR guidelines later in the chapter.

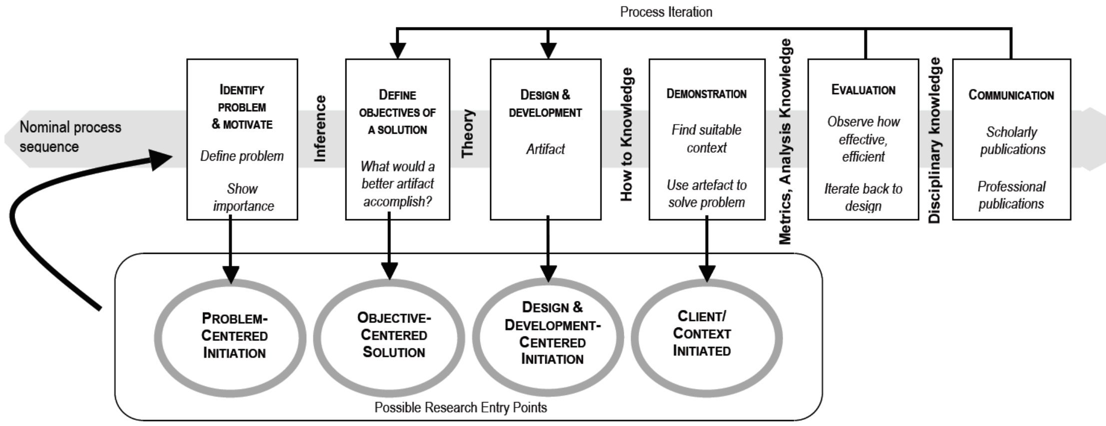

Figure 3.3 Design Science Research Methodology Process Model (Peffers, et al., 2008)

One objective of the proposed research DSR methodology is "…a nominal process model for doing IS research…" (Peffers, et al., 2008). The DSR process model consists of 6 activities that cover the entire research from motivation to commutation. Despite being designed as a set of sequential activities, the authors point out that, depending on the actual project, the *Research Entry Point* might vary (Peffers, et al., 2008). There are four different entry points defined: centred around problem (e.g. research gap in literature), objective centred (e.g. industry project), design and development (e.g. known yet not implemented artifact), and client/context (e.g. real world consulting projects). The entry point for this research is, "Problem Centred Initiation"; a problem which was identified in existing literature will be investigated in the research process. This implies that the process will be used in sequential order starting with activity 1.

3.4.1 Identify Problem and Motivate

The first activity in the DSRM process includes the definition of the problem and the justification of the "value of a solution" (Peffers, et al., 2008). There are different opinions in regards to where that motivation should stem from, Hevner et al. (2004) summarise it as “important and relevant problems”.

The focus of this research is BI and how the concept can be ‘enhanced’ to better fit with current business requirements. BI literature suggests several variations and improvements of the BI concept. Those suggestions include better integration into business processes, lower response times (Real Time BI), higher degree of automation, more flexibility and adaptivity.

Drivers for research in this area are on the one hand business (i.e. non technical) related and stem from the increasing competitiveness of business in general and the need for flexibility to react to changes in the environment (i.e. the market). On the other hand new and improved technology (e.g. concepts, software, hardware) allows the design and development of more advanced systems.

The research aims to show that Multi Agent Technology can be used to deliver business intelligence and provide a flexible architecture. In particular it is the aim to deliver decision making capabilities into the actual context of an organisation. Vaishnavi & Kuechler (2007, p. 72) note that Design Science is sometimes called “Improvement Science”. This research can be understood as such. It is the aim of this research to design a ‘enhanced’ BI concept as an improved version of BI. The claim made in this research is that the MAEBI conceptual model is an “improved instance of tool” (Vaishnavi and Kuechler, 2007).

Foremost, DSS is a core subject area to IS discipline (Burstein and Holsapple, 2008), thus research, given that it is rigorous and relevant, in this area is justified. Since DSS and BI concepts were first suggested we have witnessed significant change in technology, like the general increased adoption of technology and reduced cost of computing.

One can investigate an object under study from different viewpoints. The technologies and concepts used in this research can be analysed in very different contexts and angles. BI and DSS represent entire research fields, ranging from technical issues to psychological aspects like how users interact with the system. Agent and MAS research has left infancy but is still in a rather early stage and will require attention for years to come.

3.4.2 Define Objectives of a Solution

Whereas the first activity (in Peffers DSRM) was concerned with the general identification of the problem (gap), Activity 2 in the process requires definition of the objectives of the proposed solution.

In general the objectives of an artefact can be defined in two ways, quantitative or qualitative (Peffers, et al., 2008). Nunamaker et al. (1990-91, p. 93) argue "The advancement of IS research and practise often comes from new systems concepts.". Considering this, and the focus defined in 3.4.1, 'enhancing' the BI concept, to design a BI / MAS architecture, it is difficult to state quantitative goals for the new system. Instead, Chapter 4 describes the advantages of a Multi Agent Enhanced BI system, why it is technically feasible and how it can support organisations to create value. This is justified against literature in the respective fields.

3.4.3 Design and Development

Based on the objectives identified, the artifact is designed and developed. Peffers et al. (2008, p. 55) argue that "… a design research artifact can be any designed object in which a research contribution is embedded in the design.". During this design and development process, the objectives of the proposed solution are translated into actual features and capabilities. This process is documented and design choices made are reasoned against the literature of the respective field.

3.4.4 Demonstration (pMAEBI)

Activity 4 (in Peffers DSRM) requires demonstration of the proposed artifact. To do so, a prototype was designed and implemented based on the MAEBI concept. This follows the requirement of Nunamaker et al. (1990-91, p. 93) "Systems must be developed in order to test and measure the underlying concepts."

The prototype system, called pMAEBI (P=Pricing), is implemented in the context of retail pricing. This problem domain was chosen as it can highlight some of the features of the proposed architecture. Also, pricing by itself presents an interesting area, both from an IT/IS and business perspective.

The purpose of this protoptype is to show the feasability of the artifact, not necessarily a quantitative improvement. In this relation Vaishnavi et al. (2007, p. 21) point out "The implementation itself can be very pedestrian and need not involve novelty beyond the state-of-practice for the given artifact; the novelty is primarily in the design, not the construction of the artifact."

3.4.5 Evaluation

Hevner et al. (2004) see the evaluation of the proposed design artifact as a crucial part of a DSR process. It is argued that the evaluation requirements are defined by the business environment. Five categories of suitable methods for evolution are listed and presented in Table 3.1. Not all approaches necessarily fit a given research project and the nature of the problem or designed artefact.

Based on the options presented, the main method to evaluate the proposed MAEBI artefact is experimental in form of a simulation. The rationale for this is that Agent / MAS literature suggests that simulation evaluation and testing are appropriate methods. For example Theodoropoulos et al. (2009, p. 77) argue "Multi-agent systems (MAS) are often extremely complex and it can be difficult to formally verify their properties. As a result, design and implementation remains largely experimental, and experimental approaches are likely to remain important for the foreseeable future. Simulation is therefore the only viable method to rigorously study their properties and analyze their emergent behavior.".

In addition Helleboogh et al. (2009) quote (Himmelspach, et al., 2003) and state that simulation is a "… safe and cost-effective way for studying, evaluating and configuring …" of multi agent systems (Helleboogh, et al., 2009, p. 2).

1. Observational	Case Study – Study artifact in depth in business environment
	Field Study – Monitor use of artifact in multiple projects
2. Analytical	Static Analysis – Examine structure of artifact for static qualities (e.g., complexity)
	Architecture Analysis – Study fit of artifact into technical IS architecture
	Optimization – Demonstrate inherent optimal properties of artifact or provide optimality bounds on artifact behavior
	Dynamic Analysis – Study artifact in use for dynamic qualities (e.g., performance)
3. Experimental	Controlled Experiment – Study artifact in controlled environment for qualities (e.g., usability)
	Simulation – Execute artifact with artificial data
4. Testing	Functional (Black Box) Testing – Execute artifact interfaces to discover failures and identify defects
	Structural (White Box) Testing – Perform coverage testing of some metric (e.g., execution paths) in the artifact implementation
5. Descriptive	Informed Argument – Use information from the knowledge base (e.g., relevant research) to build a convincing argument for the artifact's utility
	Scenarios – Construct detailed scenarios around the artifact to demonstrate its utility

Table 3.1 Design Evaluation Methods (Hevner, et al., 2004, p. 83).

Using a simulation environment and artificial data seems to be the best fit. The approach does also correspond to testing and evaluation methods in MAS literature.

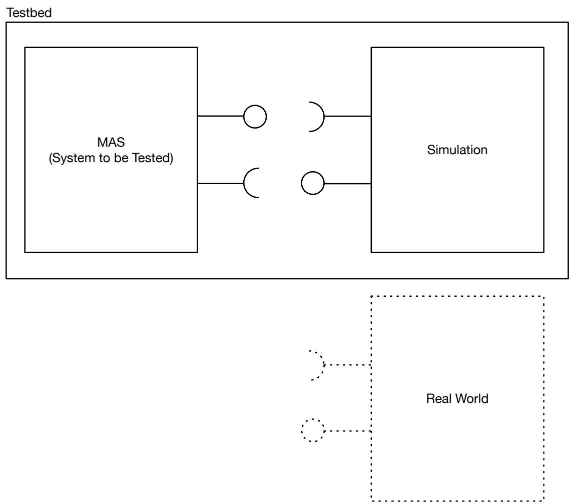

Figure 3.4 Software in the Loop Testbed

To show the relation of the proposed artifact to the business environment (Hevner, et al., 2004), the testbed design follows a software-in-the-loop architecture (Figure 3.4). This architecture models the components of the testbed as individual modules (as opposed to one 'integrated' component). Each module has interfaces similar to 'real world' systems. For example the simulation module exposes sales data which in turn is the input for the prototype (the system to be tested). This data is then processed by the prototype and fed back to the simulation (the loop). This architecture allows the researcher to conceptually 'plug-in' the new solution into an existing environment.

Section 2.3.6 and chapter 5 present technical details of the implementation and evaluation process of the artifact. The process is that the MAEBI concept is presented in Ch 4; Ch 5 presents pMAEBI system, which applies the concept to a business problem, namely pricing. The design and implementation of a simulation test-bed is described in Ch 6.

3.4.6 Communication of Research

The importance of communication of DSR research is often stressed (e.g. Hevner, et al., 2004; Peffers, et al., 2008). As this is a PhD research project, this thesis is obviously a detailed and comprehensive piece of communication.

Gregor & Hevner (2011) working on a Design Science Research Schemata to appropriately communicate DSR projects. Table 3.2 is from (Gregor and Hevner, 2011) and indicates the related chapters of this thesis.

Section	**Contents**	**Thesis Chapter**
Introduction	*Problem definition, problem significance/motivation, introduction to key concepts, research questions/objectives, scope of study, overview of methods and findings, theoretical and practical significance, structure of remainder of paper.* For DSR the contents are similar, but the problem definition and research objectives should specify the ***goals*** that are required of the artifact to be developed. The ***relevance*** of the research problem must be clearly stated.	*Chapter 1*
Literature Review	*Prior work that is relevant to the study, including theories, empirical research studies and findings/reports from practice.* For DSR work, the prior literature surveyed should include any prior design theory/knowledge relating to the problem to be addressed, including artifacts that have already been developed to solve similar problems. An aim is to show the "gap" that is still to be filled. Reference should also be made to the ***justificatory (kernel) theory*** that informed the design of the new artifact. A fuller explanation of the justificatory theory may be better placed in the Artifact Description section, matched with the specific artifact component to which it applies. However, it may help to signal what is to come by giving a brief description of the justificatory theory here.	*Chapter 2*
Method	*The research approach that was employed.* For DSR work the specific DSR approach adopted should be explained, with reference to existing authorities (for example, Hevner et al. 2004; Nunamaker et al. 1990-91; Peffers et al. 2008). ***Research rigor*** must be clearly demonstrated in selection of methods and techniques for the building and evaluating of the artifact.	*Chapter 3*
Artifact Description	This section (or sections) should occupy the major part of the paper. The format is likely to be variable but should include at least the description of the ***design artifact*** and, perhaps, the ***design search process***. If the aim is to show a design theory, this section should include ***meta-requirements***, ***constructs***, any ***instantiation, principles of form and function, artifact mutability*** and ***principles of implementation***. ***Justificatory knowledge*** for the nature of the	Chapter 4

	artifact may also be provided.	
Evaluation	The artifact is evaluated to demonstrate its worth with evidence of utility, quality, and efficacy. A rigorous ***design evaluation*** may draw from many potential techniques, such as analytics, case studies, experiments or simulations (see Hevner et al. (2004)).	Chapter 5
Discussion	*Interpretation of the results: what the results mean and how they relate back to the objectives stated in the Introduction Section. Can include: summary of what was learned, comparison with prior work, limitations, theoretical significance, practical significance, areas requiring further work.* ***Research contributions*** are highlighted and the broad implications of the paper's results to research and practice are discussed. A summary of what has been learned could be provided by expressing the design theory (if any) produced in terms of the design theory components specified by Gregor and Jones (2007). The generality of the design theory can be expressed in terms of ***testable propositions***. Claims for novelty and utility should be expressed as well as claims for a contribution to design theory if appropriate.	*Chapter 6*
Conclusions	*Concluding paragraphs that restate the important findings of the work.* States the main ideas in the contribution and why they are important.	*Chapter 7*

Table 3.2 DSR Publication Schema (Gregor and Hevner, 2011 (Working Paper))

3.5 Research Methodology Validation

Research rigor is a fundamental aspect of DSR in IS. The research methodology developed in this chapter is summarised and evaluated against the 7 DSR guidelines suggested by Hevner et al. (2004)

3.5.1 Guideline 1 – Design as an Artifact

Guideline 1 requires that the output of a DSR project is some form of artifact. The aim of this research is to 'enhance' the BI concept with agent technology. The resulting MAEBI concept, the artefact, builds upon existing research and addresses gaps identified. The design processes and justification of (design) choices are documented.

In this relation it should be noted what an artifact is and what not. Hevner et al. (2004) refer to Tsichritzis (1997) and Denning (1997):

"Furthermore, artifacts constructed in design-science research are rarely full-grown information systems that are used in practice. Instead, artifacts are innovations that define the ideas, practices, technical capabilities, and products through which the analysis, design, implementation, and use of information systems can be effectively and efficiently accomplished."

3.5.2 Guideline 2 – Problem Relevance

Guideline 2 stresses the significance of relevance in DSR research. The proposed MAEBI concept is a form of a decision support tool. Computational decision support is not just core to the IS discipline but can also have a significant impact on the performance of an organisation (Arnott and Pervan, 2008; Burstein and Holsapple, 2008).

The MAEBI concept is not aimed at a specific problem domain, it rather focuses on flexibility to be applicable in many different problem domains.

In context of the evaluation of the MAEBI concept the aim was to find a problem domain that showcases some of the functionality of the artefact and is also a relevant and significant business problem. Pricing, in particular retail pricing, presents such a significant problem to businesses (Boyd, 2007; Phillips, 2005). Retailer's (e.g. supermarkets) have to set prices for many thousands products and have to incorporate many different factors, like costs, consumer preferences and competition.

3.5.3 Guideline 3 – Design Evaluation

Hevner et al. (2004) emphasises the importance of rigorous evaluation of an artefact. They further suggest several evaluation methods that are applicable in IS DSR research. The context of the evaluation is given by the business environment and the proposed artifact has to integrate into the environment.

Of the suggested evaluation methods simulation with artificial data was chosen as this method also fits with suggestions from Multi Agent related literature. The testbed will simulate artificial consumers that shop at different retail stores, which use either a MAEBI system for pricing or a 'traditional' centralised system. This allows a comparison between the approaches.

3.5.4 Guideline 4 – Research Contributions

According to Hevner et al. (2004) there are three different types of contributions that can emerge out of a DSR project. Those contributions can be, the design artefact itself, addition(s) to the Foundations of DSR or Methodologies.

Based on the reviewed literature in decision support systems and multi agent systems, opportunities were identified to improve and advance the concept of BI by combining the two technologies. The contribution of this research is the artefact itself to "… apply existing knowledge in new and innovative ways." (Hevner, et al., 2004, p. 87). The contribution is discussed in more detail in 1.5 and 6.2.4.

3.5.5 Guideline 5 – Research Rigor

Hevner et al. (2004, p. 87) argue that DSR requires "… the application of rigorous methods in both the construction and evaluation of the designed artifact.".

Design and construction of the artefact, the MAEBI framework, is described in chapter 4. The framework is based on relevant literature in the fields of DSS/BI and Agent/MAS.

To evaluate and show feasibility of the MAEBI framework, the framework was used to implement the pMAEBI system (p = pricing) in combination with a simulation system. Evaluation by simulation is an accepted method in the respective reference disciplines (DSR and Agent Literature).

3.5.6 Guideline 6 - Design as a Search Process

Design science is an iterative process to find an effective solution to a problem. This involves the use of the knowledge base in the respective reference disciplines.

Chapter 2 presents a selection of literature that was reviewed for this research and presents the knowledge base for this research. Chapter 4 continues the search process by deriving design objectives from literature. These objectives are transformed into an actual design solution (MAEBI framework).

In chapter 5 this framework is implemented using suitable technologies that illustrate its applicability in a problem domain.

Each step requires the search for appropriate means to reach desired ends.

3.5.7 Guideline 7 - Communication of Research

The results of design science research projects are interesting and relevant for both management and technology-oriented audiences. However those audiences have different perspectives and information needs. Hevner et al. (2004, p. 83) write "Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences." .

As this research is a PhD project, this thesis is the main piece of communication and targeted towards an academic audience.

Guideline	Description	Mapping
Guideline 1: Design as an Artifact	Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.	MAEBI concept (pMAEBI implementation)
Guideline 2: Problem Relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.	enhancing Business Intelligence
Guideline 3: Design Evaluation	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.	pMAEBI prototype (simulation/testbed)
Guideline 4: Research Contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.	Artefact that combines two complementing technologies to improve business decision making.

Guideline 5: Research Rigor	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.	Methodology based on established literature
Guideline 6: Design as a Search Process	The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.	Literature Review in context of BI, MAS, DM/KD, Pricing
Guideline 7: Communication of Research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.	Thesis & Publications

Table 3.3 DSR Summary

3.6 Chapter Summary

The purpose of this chapter was to develop and justify a methodology for this research. To do this the Research Pyramid was used as a high level framework.

DSR is the paradigm chosen for this research and is according to the presented literature (e.g. Hevner) a suitable approach to investigate problems in the domains of IT and IS. The problem under investigation is, how to enhance BI, is an important and relevant IS area.

The paradigm, in combination with Peffers's DSRM research process model, outlines the individual research activities and ensures a rigorous research process.

Simulation and Software-in-the-loop testing are approaches that are described in literature (see chapter 2) and are suitable, in particular, considering the experimental nature of MAS systems.

In summary, using the research pyramid as guideline to establish a research methodology, in combination with Hevner's DSR guidelines and Peffers's DSRM process model present a complete research methodology to address the research question.

Chapter 4 - Multi Agent Enhanced Business Intelligence (MAEBI)

4.1 Introduction

This chapter reiterates the design objectives of the proposed solution and describes the design process of the Multi Agent Enhanced Business Intelligence - MAEBI concept (the artefact). The chapter corresponds to activity 2 (define objective of a solution) and activity 3 (design and development) of Peffers DSRM model (see chapter 3).

Peffers et al. (2008) describe activity 2 as "Define the objectives for a solution. Infer the objectives of a solution from the problem definition and knowledge of what is possible and feasible." This definition can be either quantitative or qualitative; the artifact here will be defined qualitatively. They state that for this activity knowledge of the current state and current solutions is required. Section 4.2 will outline the problem / solution space and describe benefits of the system's respective attributes.

Activity 3 in Peffers DSR process entails the actual design and creation of the artifact. This means that an artefact has to be designed that solves the problem and addresses the established objectives in context of the environment (e.g. business). Peffers et al. (2008, p. 13) write "This activity includes determining the artifact's desired functionality and its architecture and then creating the actual artifact.". To do so, knowledge about reference disciplines and technologies is required. Reference disciplines in this research are BI and agent and multi agent systems.

To conclude the chapter a synthetic case study is presented that highlights some of the functionality of the MAEBI system. It is also compared with and differentiated from other approaches and concepts to crystallise the "enhancement" over traditional BI.

4.2 MAEBI Design Objectives

Decision Support Systems have to develop over time to reflect the changing environment they operate in (O'Leary, 2008). This change can occur in technical areas, like new and improved database systems, new communication technologies or organisational change like shifting user needs, new and more complex problems or problem understanding and similar. Barone et al. (2010) stated in a recent paper that BI systems are still inflexible and do not support businesses to their fullest capabilities. It is further argued that data is generally available (e.g. it is captured), however, it is still difficult to put that data into "meaningful and productive" use.

The purpose of BI systems is, to varying degrees, to support organisations in their decision-making process and this remains the overall objective of the proposed system. However the new concept may better reflect some of today's demands, by providing a flexible and dynamic architecture to deliver decision-making (as opposed to decision support) capabilities throughout an organisation. Most importantly the design should allow organisations to capture local (market) characteristics and be able to address such.

To achieve this, different trends and suggestions were identified in the literature review; this was distilled to five objectives (supporting decision process, real-time BI, localised, adaptive, automation) , which are considered during the design stage of the MAEBI architecture. These objectives or characteristics are motivated by technological change and reflect industry and business trends.

4.2.1 Objective 1: Supporting the Decision Process

The proposed system is designed based on the BI concept. BI in turn is a form of a decision support approach and, as such, supports the decision process. There is no explicit or implicit ranking or valuation amongst the objectives, however it is apparent that a sound support of the underlying process is of major importance.

As established in chapter 2 decision making is a complex process and, considering today's business environment, this complexity will continue to increase. This complexity presents itself as an increase in data sources and data volume, change in the environment (e.g. customer behaviour, government regulations, financial) and consequently that cause – effect relations are more difficult to identify (Cassaigne and Lorimier, 2006; Hall, 2008).

The benefits of a DSS system can either be in a better decision process and/or in a better decision outcome (Pick, 2008) and "… relaxes cognitive, temporal, spatial and/or economic limits on the decision maker." (Holsapple, 2008b, p. 163). Advantages here refer to the situation where the decision maker has the support of a DSS vs. the case where the decision is made solely by the human participant.

MAEBI is designed with Boyd's "Observe, Orient, Decide and Act Loop" (OODA) (e.g. Carlsson and Sawy, 2008; Haas, Mills, and Grimaila, 2011) (Figure 4.1) in mind as underlying framework. The model was first used by John Boyd (US Air Force Pilots) to describe and explain the decision making process of fighter pilots. The model found its application not just in military decision making but also in business and DSS systems (Haas, et al., 2011; Negash and Gray, 2008). The purpose of the OODA loop in the MAEBI framework is to guide the automated decision making.

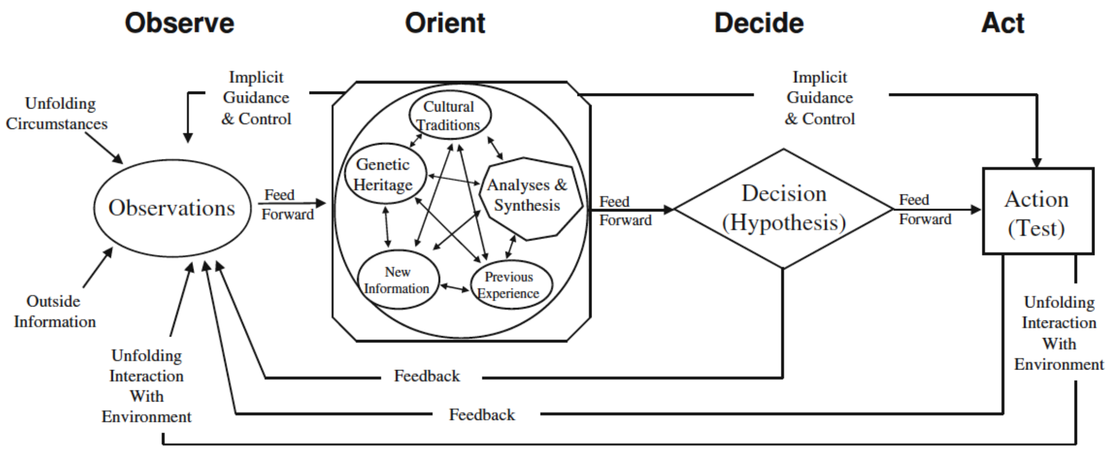

Figure 4.1 Boyd's OODA Loop (Haas, et al., 2011, p. 178)

The general “flow” of this loop is that the decision maker (or the software in case of this research) gathers data and information (Observer), puts the information into context of the current situation (Orient), makes a decision based on the evaluation of the “situation” identified in the Orient stage and finally implements this decision (Act). As the approach was design around human fighter pilots, there are no strict quantitative models that describe each stage or the transition between the stages. The feedback lines indicate that at every stage the “is” situation is compared with the “expected” situation and underlines the dynamics of the system. This process is adapted in this research to guide the decision process in the agents. Instead of having a “human” evaluation and decision methods, the software version uses statistical and/or AI methods to analyse the data and base decision based on the data. The feedback channels indicate that the system can learn from it’s own behaviour and that the behaviour of the system also become input data as part of the environment (for example to identify/prevent decision bias). The stages in more detail:

- Observe. The first stage monitors the environment and gathers data from various sources. This requires 1) access to data sources 2) communication facilities 3) transformation capabilities 4) data storage

- Orient. The second stage is concerned with the organisation and sense making of the data acquired previously. To do this data mining (DM) and knowledge discovery (KD) techniques have to be available and accessible for the system. The focus here is on unsupervised methods (Holsapple, Jacob, Pakath, and Zaveri, 2008)

- Decide. Stage three is concerned with choosing one alternative out of the solutions generated in stage 2. This selection process can be implemented in different ways, DM algorithms can be used or rule based systems can make a selection.

- Act. Finally the decision/choice made in the previous step has to be implemented. To allow the system to implement a decision, it needs access to operational systems.

4.2.2 Objective 2: Real Time BI

Gartner Research was one of the first to introduce the term Zero-Latency-Enterprise (ZLE) (Schulte, 1998). The idea of ZLE is to reduce the time between a business event and the appropriate action (Figure 4.2) to improve business performance. The idea triggered several developments in BI and DSS. Terms like Active Warehousing, Real-time Analytics, Real-time Warehousing, Real-time Decision Support and similar appeared in the literature (e.g. Nguyen, Schiefer, and Tjoa, 2005; Vahidov and Kersten, 2004).

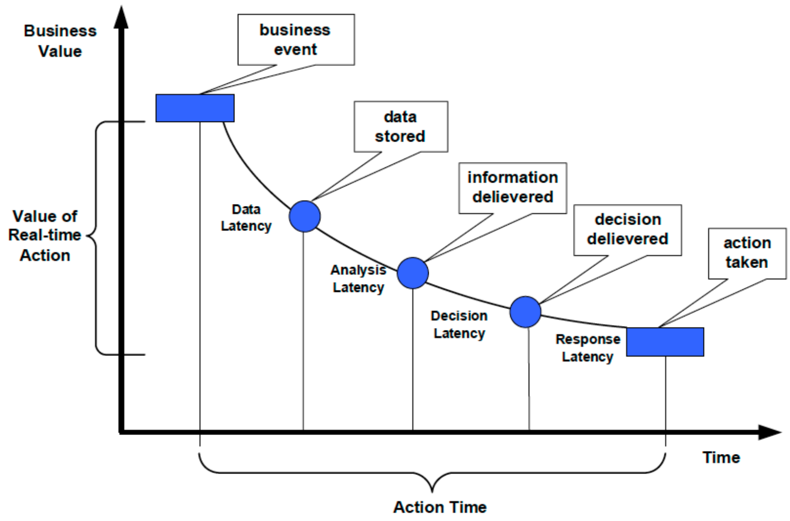

Figure 4.2 Zero-Latency-Enterprise adopted from (Nguyen and Tjoa, 2006, p. 168)

The term 'real-time' is often used, but what real-time means in a specific context is rarely defined. Depending on the situation and the perspective on that situation, time scales change. For example, a CEO thinks in 5 year terms whereas a production worker thinks in days or hours. IT/IS systems are often adjusted to a "human schedule" for example "backup on weekends". Sometimes the term 'right-time' is used as an alternative to 'real-time' to emphasise an implicit relevant time scale for a problem domain.

Azvine et al. (2006) acknowledge the lack of an accurate definition or understanding of 'real-time' and suggest three different 'usages' or meanings:

- "Zero Latency" Processes
- Up To Date Information whenever needed by user (e.g. manager or other process)
- KPIs relate to current (i.e. now) situation

The authors give two reasons for the importance of real-time BI (RT-BI), 1) the business environment and 2) advances in technology. The environment businesses operate in changes rapidly (e.g. share prices, sales pattern etc.) and a continuous flow of information is required as opposed to pre-schedule reports (e.g. daily, weekly etc.). Today's technology would generally allow the design of RT-BI systems. In particular the Internet is mentioned as a means of distributing data throughout an organisation. However current BI systems face two challenges in regards to providing RT-BI i.e. the transition from data to information and from information into action. The transition from data to information is challenging because highly skilled professionals are required (e.g. expensive, limited availability). The transition from information into action is currently "manual". BI systems provide data and reports but these outputs are not automatically applied to the respective processes. The prospect of RT-BI is a seamless flow from sensing data to adjusting business process.

Meredith et al. (2008) report that, in particular the ETL processes are often scheduled to run overnight, on weekends or other time of low system load. This means that the DW is always (to varying degrees) out of date, a state that is not desirable in particular in RT-BI systems. Real-Time refers to different timescales and may vary significantly. For example if a RT-BI system is used to support a decision that is due once a month there should be sufficient time to run all ETL processes. For a RT-BI system that is used in a financial trading environment on the other hand, it is unlikely that an "every-night" schedule is sufficient for the purpose.

The term "Operational BI" is emerging in relation to RT-BI and describes a paradigm shift or extension of BI from a traditionally rather tactical or strategic concept to an operational (real time) one. Being able to access operational systems in real time, allows that BI can be applied to problems across all levels (strategic, tactical, operational) of an organisation (Anderson-Lehman, Watson, Wixom, and Hoffer, 2008).

It can be concluded that operational data is usually accessible and technology is generally available. Being able to use DSS/BI in a real time to support operational decision making can show new potential (e.g. Marsden, 2008; Meredith, et al., 2008).

In the context of this research and for the proposed MAEBI system, the following properties are deemed necessary to facilitate real-time BI functionality:

- access to a variety of data sources
  (e.g. internal ERP, POS ; external Supplier)
- access to processes (Process Management Systems)
- Extract Transform Load (ETL) capabilities
- Data Storage (Database System)

4.2.3 Objective 3: Localised

IT systems today often follow a client server architecture, that is a central server and clients that, using some form of network connection, connect to that server (e.g. using SOA). In particular businesses (e.g. retail chains) have a "similar architecture" in respect that there is one headquarters managing a number of subsidiaries or outlets. The idea behind this approach is economies of scale, to achieve a competitive advantage by producing/buying in large quantities to reduce unit costs.

The problem with this approach, however, is that those companies, sometimes completely, may ignore local market characteristics and subsequently miss profit opportunities. Rigby and Vishwanath (2006) summarise current developments as "For a quarter century, the big winners in consumer markets have pursued strategies of standardization. But success for retailers and product manufacturers now hinges on their ability to cater to local differences – while maintaining scale efficiencies."

The business mantra “Think Global – Act Local” seems desirable and feasible, but it turns out that this is complex. Rigby et al. (2006) list two reasons, one is that on a local level the required skill set is not available (e.g. store manager is not a statistician) and secondly the risk of adjusting “too much” to local characteristics and by that introducing uneconomic complexity into the business. However Rigby et al. (2006) further argue that “sophisticated data analysis” combined with “innovative organisational structures” can help businesses to gain a competitive advantage. By leveraging local data and data analysis methods, businesses gain (new) insights about their customers. Such insight is hard to copy for competitors.

Trivedi (2011) describes that current practices in consumer consumption research are based on data that is usually collected at house hold level or store level and then aggregated in some form, often in combination with some form of regression. This leads to the situation that some consumption / behaviour patterns might be unobservable on an aggregate level. To add to Trivedi (2011), one can raise the question, why do we gather and store transactional data on a local level if that data is not fully leveraged.

A flexible, automated and localised decision making system would address exactly such business problems. Such a system would not require significant (if any) human interaction (e.g. by the store manager) and simultaneously the system would utilise local transactional data. In other words, available data would be better leveraged in the decision process to better serve customers by learning from him/her.

4.2.4 Objective 4: Adaptive

Organisations and processes change and DSS/BI systems have to adapt to their changing environments. The issue of adaptiveness in decision support is discussed for example by (Azvine, et al., 2006; Holsapple, et al., 2008; Michalewicz, et al., 2007). Michalewicz at al. (2007) and Michalewicz et al. (2008) discuss the topic of adaptive business intelligence in more detail. The motivation behind their concept is again the fact that data is generally available in organisations but not fully utilised. Current BI systems are described as "… responsible for collecting and digesting data, and presenting knowledge in a friendly way …". The concept of Adaptive BI was built around three objectives, prediction, ability to adapt and take appropriate actions (Michalewicz and Michalewicz, 2008). These objectives were derived from Fogel et al. (1966) that intelligence entails the ability to predict, the ability to adapt and the ability to take appropriate actions (Michalewicz and Michalewicz, 2008). Those objectives here refer to the problem domain or the problem environment.

Figure 4.3 illustrates the conceptual flow of data in current BI systems. Data is gathered (e.g. from operational systems) and refined (e.g. ETL) into information. Applying data mining on the available information may lead to deeper insights (knowledge). According to the authors just providing knowledge is not sufficient and systems should suggest the best course of action. The proposed Adaptive BI concept, illustrated in Figure 4.4, builds on the BI process and adds prediction and optimisation steps. This allows further analysis of the available knowledge and recommend action of "… the best course of action (based on past data) …" (Michalewicz and Michalewicz, 2008, p. 57).

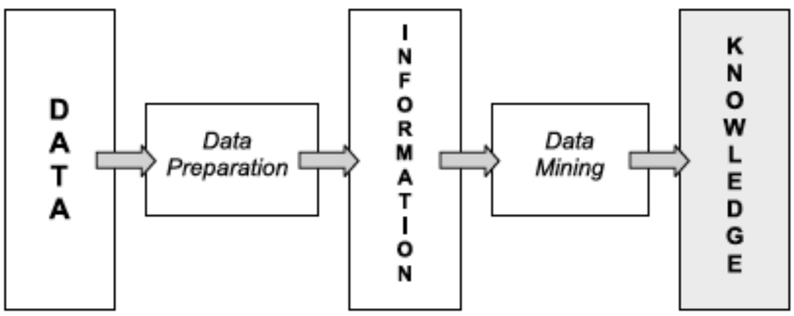

Figure 4.3 BI Process (Michalewicz, et al., 2007, p. 4)

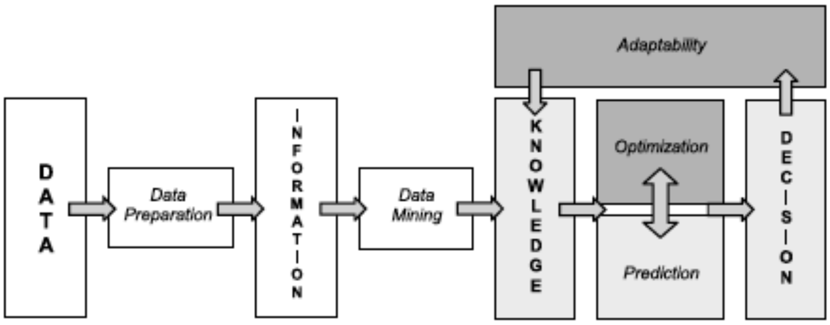

Figure 4.4 Adaptive BI Process (Michalewicz, et al., 2007, p. 5)

The "Adaptability" module allows the system to learn from previous decisions/recommendations and can improve future recommendations (e.g. decision bias).

Holsapple et al. (2008) discuss the development of an adaptive DSS. According to the authors the point of differentiation is whether such a system uses un-supervised as opposed to supervised learning techniques.

Adaptive BI is arguably a valuable development in BI and promises to utilise data and technology (i.e. data mining & machine learning) better and subsequently better support decision makers. The proposed MAEBI framework is not limited to specific data mining and/or machine learning algorithms. Implementation requirements will influence such a decision. For the prototype system an artificial neural network algorithm was used (see 5.6.6).

4.2.5 Objective 5: Automation

Traditionally DSS and BI are tools that focus on the support of the decision maker. For example Holsapple (2008a) writes "… get the right knowledge to the right decision makers at the right times in the right representations at the right costs." However the increase of available data and incr methods for testing of agent based systems.

easingly complex decisions might cause cognitive overload of decision makers (Vahidov and Kersten, 2004).

Literature frequently uses the term automation in context of BI and DSS, nonetheless a definition or classification of the degree of automation is commonly missing. Cummings (2004) presents a classification scheme of levels of automation (Figure 4.5).

Automation Level	Automation Description
1	The computer offers no assistance: human must take all decision and actions.
2	The computer offers a complete set of decision/action alternatives, or
3	narrows the selection down to a few, or
4	suggests one alternative, and
5	executes that suggestion if the human approves, or
6	allows the human a restricted time to veto before automatic execution, or
7	executes automatically, then necessarily informs humans, and
8	informs the human only if asked, or
9	informs the human only if it, the computer, decides to.
10	The computer decides everything and acts autonomously, ignoring the human.

Figure 4.5 - Automation Levels from (Cummings, 2004, p. 2)

Parasuraman and Sheridan (2000, p. 286) argue that “Technical developments in computer hardware and software now make it possible to introduce automation into virtually all aspects of human-machine systems”.

Without trying to determine where exactly current systems are, literature mentions the increasing pressure on decision makers to handle more data and more complex decisions in a shorter period of time (e.g. Phillips-Wren and Jain, 2007; Sargut and McGrath, 2011). New systems should consider this and aim for high degrees of automation to relieve the decision maker with the goal of better decision outcomes.

Better can refer to different metrics and depends on the situation/context. For example in the prototype system, better refers to a higher store profits.

4.2.6 Design Objectives Summary & Research Gap

Decision support systems (including BI) have to evolve over time (O'Leary, 2008). DSS systems are complex systems that are influenced by people (users), organisations and technology. If this environment changes, DSS tools have to adjust to that new situation. The design objectives identified (sections 4.2.1 – 4.2.5) describe such changes as noted in literature, which are not or only partially reflected in current BI systems.

The motivation behind this research is to help to evolve BI by extending current technologies to better support businesses in their decision making. The framework itself does not focus on a particular problem domain, however it focuses on problem environments where a local perspective on decision making may differ from a global view.

Activity 2 in Peffers et al. (2008) DSR process entailed the definition of the design objectives (see list below) of the new artefact. This section established those in detail and are summarised below. Those five design objectives will be transformed into features and systems characteristics of the MAEBI system (Activity 3) in the next section.

- **Flexibility**
  The proposed system has to be flexible to adjust to complex problems, organisational structures and business processes. If this environment changes, the system has to change accordingly.

- **Automated**

  Traditionally DSS and BI are support systems as opposed to decision-making systems. The increase in the complexity, the required timeliness of decisions and the number of decisions create a necessity to automate, that is make and implement a decision without human interaction.

- **Localised**

  Organisations grow and while economies of scale remains an important success factor, so does customer focus. It is important for companies to address customer needs and adjust to local market characteristics to improve profitability.

- **Adaptive**

  The (business) environment is constantly changing and the system has to adjust to this change.

- **Real-Time (Right-Time)**

  As such a system is embedded or interacts with the business process, the process dictates the speed and the system has to produce information at this speed.

4.3 Multi Agent Enhanced Business Intelligence (MAEBI)

4.3.1 Introduction

The last section outlined the design objectives of the new MAEBI concept. This section continues the research process by transforming the objectives into features and characteristics and defines the design of the artefact (Activity 3 in Peffers et al. (2008)).

MAEBI in a general sense is a design framework and has similar goals to every BI/DSS system, to support organisational decision making. More specifically, MAEBI is designed as an enhanced version of BI. Enhanced refers to a more flexible form of BI. This means that the MAEBI concept, like BI, is a data centric DSS. There are two major differences between traditional BI and MAEBI, 1) MAEBI has a decentralised focus and 2) MAEBI includes decision execution and is not limited to supporting (e.g. reporting) functionality.

The following section first discusses agent and multi agent technology as the means to realise the MAEBI system followed by a description of the MAEBI concept itself. MAEBI consist of two Agent Types, a so called Decision Unit (DU) and a Configuration Engine (CE). The former is the central element of the system and is responsible for the local decision making, the latter has administrative and maintenance functions in the system. Hevner et al. (2004, p. 82) state "It [the artefact] must be described effectively, enabling its implementation and application in an appropriate domain." To do so, each agent type is explained in detail in regards to purpose and functionality and suggestions are made as to how such functionality can be implemented.

4.3.2 Agent & Multi Agent System

Agent and Multi Agent Systems have the attributes of being highly flexible and dynamic (e.g. Kirn, 2006). The agent design paradigm of breaking functionality into autonomous, to some extent intelligent, agents is promising. Lim and Jain (2010) see MAS systems as a promising choice for intelligent decision making systems. DSS and BI are complex systems with many (sub) systems or modules and might be specialised for different application areas. This complexity opens a range of opportunities to apply the agent paradigm in DSS/BI systems.

The flexibility and versatility of the agent metaphor allows it to be used in different ways in a BI system. 1) The Agent paradigm can be adopted to develop the actual software and replace for example object oriented languages. For example, the database management system can be developed using an agent programming language instead of an OOP language. An example for this would be CouchDB[5] an open source database system that is primarily developed using the Erlang[6] programming language. Erlang is a programming language / runtime that follows the actor model and allows the effective implementation of concurrent applications. This characteristic translates to the scalability of CouchDB. The Erlang/CouchDB combination does not show all aspects of agent based software development but it does show the applicability in highly concurrent applications.

2) Another application of the agent metaphor is to develop and facilitate (machine) intelligence (e.g. DM / KD) aspects in a BI system. For example Cao (2009) discusses not just the application of agents in data mining but the mutual interaction between data mining and agents in book length. This means that agents can be used to implement actual data mining methods and data mining methods can facilitate agent intelligence (agent behaviour, agent learning).
Similarly Zhang and Zhang (2004) suggest their idea of Agent based Hybrid Intelligent Systems. The motivation behind the concept is that decisions (i.e. solving complex problems) have become more complex. Foreign exchange trading and knowledge discovery from large/multiple databases are mentioned as example problem domains. In respect to the analysis and data mining techniques employed to "solve" such problems, one method is rarely sufficient to achieve a satisfying

[5] http://couchdb.apache.org/

[6] http://www.erlang.org/

result. To address this problem the authors suggest systems that combine different analysis methods and name this concept Hybrid Intelligent Systems (Z. Zhang and Zhang, 2004). It is argued that the design of such systems is complex because they consist of a large number of different components that have to interact and agents would facilitate this.

3) A third way to use agents is as a means of distributed software architecture. Agent systems are inherently distributed in the sense that agents are individual entities. Other distributed software architecture concepts like Common Object Request Broker Architecture (CORBA) or Service Oriented Architecture (SOA) imply that every part of the system has the same "goal", whereas individual agents can have "their goals" and are independent entities (e.g.Wooldridge, 2001). Georgeff (2009) in particular, addresses advantages of agents over SOA. He argues that agent can add value in three areas, loosely coupled processes instead of "just services" (goal oriented linking of processes at run-time), context dependence (agents can decide if a process/plan is applicable in a given situation) and robustness (agent can "just try again").

4.3.3 MAEBI Components

To realise the design objectives, MEABI follows a modular architecture that is based on two agent types. The Decision Unit (DU) represents the core of the architecture and a Configuration Engine (CE) for admiration purposes. The following sections present the agent types and functionality in detail.

4.3.3.1 Decision Unit (DU)

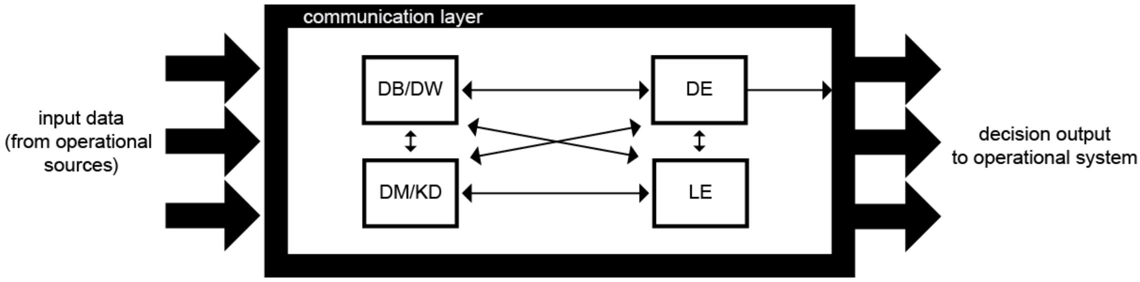

Figure 4.6 Decision Unit (DU) Overview

4.3.3.1.1 Database / Data Warehouse (DB/DW)

Like in the traditional BI concept MAEBI is also a data centric system and some form of data storage is required. The terms database (DB) and data warehouse (DW) are used interchangeably here. Both refer to a general data storage that can store operational as well as non-operational data.

In a traditional BI system, a central DW is used, which is filled with data, usually in pre-defined intervals by an Extract, Transform and Load (ETL) process (Meredith, et al., 2008). Contrary to traditional BI systems the scope of the data in a DU is local and not global. This means that the functionality of a DW, in a DU, does not necessarily change; however each DU has to be equipped with its own DW that reflects the local environment.

The data storage in the DU (the DW) is used to store different types of data that are required to make the agent function and provides data and information that represent the local environment.

Decision Unit Meta Data - Similar to the belief set in an agent, there must be some local parameters that describe the agent (the DU). This might be something trivial as an ID, or complex sets of parameters that contain login information to other systems.

Data of the problem domain – This is likely to be the most significant part of the data that a DU has to handle. This is the data source for all decision making activities within the DU. How exactly that data is represented depends on the DU's environment and the business problem. For example a DU may be concerned with the ordering of raw material for a plant. Domain data here would refer to data about the production schedule, bill of material, supplier information and similar data.

Learning - One aspect of a DU is the ability to learn and to adjust to its environment. Results of data mining activities and implemented decisions need to be stored for later comparison.

Other Temporary Data – There might be other data that is generated during the lifetime of a DU, like intermediate results in calculations or similar and it might be easier or more efficient to store those in the DB.

Data storage capabilities can be implemented in many different ways and the choice of technology depends on the actual situation (i.e. implementation). Data storage is a rather broad term, but the type of data that a DU has to handle can be very different, thus different technologies may be implemented for different data types in a DU. It is likely that some type of relational SQL database system, like MS SQL Server, MySQL, Oracle or IBM will be used. Relational databases are very flexible and there is significant knowledge and experience in practice and academia. Besides that, most businesses are likely to use such products/systems already.

Recently we have witnessed the adoption of so called "NoSQL" databases. This type of database is usually referred to as structured storage in academia and has been known for some time. NoSQL databases became particularly popular in "Web 2.0" applications that have to handle significant amounts of data, usually in a distributed environment. The advantage of NoSQL databases in comparison to their traditional SQL counterparts is that they are not relational and can better handle unstructured data (Leavitt, 2010). DSS and BI are generally implemented using relational SQL database systems. This ‘new’ approach of database system might be beneficial for unstructured data or as ‘memories’ for agents.

4.3.3.1.2 Data Mining / Knowledge Discovery (DM/KD)

BI systems as opposed to traditional DSS systems usually include advanced analytics and data mining capabilities. Such capabilities have become more important because (business) decisions have become more complex. Davenport and Harris (2007) stress the general importance of analytics in their book “Competing on Analytics” from a business perspective.

The purpose of a DU within the MAEBI system is to sense its environment by gathering data from its environment (e.g. from a business process or corporate meta data) analyse this data and, if required, adjust the process. Negash et al. (2008, p. 179) write in this relation “Analytics are the input to human and automated decision making”. Within the DU the DM/KD module is responsible for providing different analysis methods. Data Mining and Knowledge Discovery are the terms that describe a set of different techniques that allow for example Classification, Prediction and Regression (e.g. Khan, Ganguly, and Gupta, 2008). Specific implementations depend on different factors. Peng et al. (2006) point out that there are many different systems and methods suggested in literature for various DM/KD tasks.

MAEBI is not focused on a particular set of problems and DM/KD methods may vary in actual implementation. On a more general note, Holsapple et al. (2008) discusse the application of unsupervised learning methods for adaptive DSS systems as opposed to supervised methods for “traditional” DSS systems. Adaptiveness and automation are two of the design objectives of the MAEBI framework and unsupervised methods and techniques contribute to these goals.

Implementation can vary significantly depending on the available data and problem domain. There are many mining algorithms known in academia that can be adapted in custom applications or open source and proprietary systems (e.g. SPSS, R, MS SSAS, SAS) can be integrated. It is outside the scope of this research/thesis to give guidance in regards to selection criteria/process of DM/KD algorithms. This heavily depends on a specific requirements of a implementation. Section 5.6.6. explains the choice of algorithm for the test implementation.

4.3.3.1.3 Decision Execution (DE)

DSS systems and BI are decision support tools, they support a human decision maker to analyse data and provide functionality in the decision making process. The ultimate decision to implement a choice (the result or outcome of the process) is up to the decision maker and not to the software. Considering that the focus was on strategic and tactical decisions this was suitable, however for real-time/operational use the system must be capable of implementing a decision. In MAEBI a Decision Execution component in a DU has access to the respective operational systems that control or influence the problem domain, and can change (or adjust) this process (Figure 4.7).

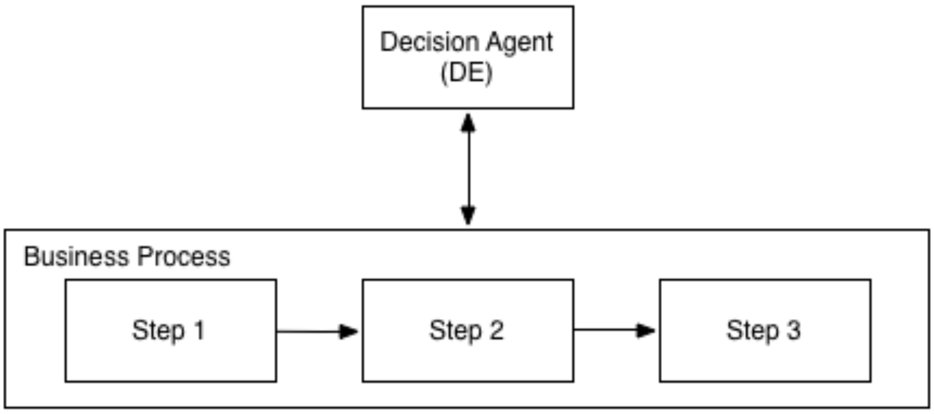

Figure 4.7 Decision Execution

This functionality is similar to what Azvine et al. (2006) describe as "RT-BI". They argue that data analysis has to be performed in real-time but also that the response has to be implemented in real-time. Requirements for such new real-time tools exceed what is currently known as Business Activity Monitoring (BAM). BAM does integrate data from different sources in real-time but only provides (presents) the information in form of RT-Dashboards to decision makers and does not usually extend to an automated solution.

The DE module can utilise the communication facilities provided by the DU to access operational systems (e.g. data base, parameters of machinery or process management systems). This allows the module to get new data and implement changes to the processes and operational systems. Depending on the application this DE module might just write a value into a table. Other implementation scenarios may require the inclusion of a rule based system to ensure that only valid data gets transmitted. Valid refers to decisions that are in line with organisational policies and/or legal requirements.

4.3.3.1.4 Learning (Feedback)

Part of Boyd's OODA loop as well as the Adaptive BI concept by Michalewicz et al. (2007), is that a system should learn from its own action. This means that if a DU has implemented a decision, the "result" or impact on the environment has to be tracked and considered as part of the knowledge of a DU. For example, a DU decides that for the next period a small can of coke will be priced at 99 cents, and it (the DU) expects that 1000 units will be sold, but after the time period only 500 units have been sold. This becomes new knowledge for the DU and presents an input in future decisions.

In the OODA loop (Figure 4.1) those feedback channels link every stage back to the Observe stage. To allow a DU to learn from previous situations and behaviour, decisions (outcomes) can be logged in the DB/DW and can be used by the DM/KD module.

The learning module of the DU primarily logs the actions and decisions of the DU and stores these in the DB/DW to be accessible for later use. This data is then used as an additional input (it becomes one aspect of the environment). It can be accessed later and compared, for example to derive information about how the environment reacts to a decision. The learning process might be complex in itself. In such cases the learning module can use the DM/KD module to analyse data.

4.3.3.1.5 Communication

The ability to communicate is not just central to MAS systems, but also for the MAEBI system. It is the "glue" between the modules of the DU and its environment. For example, there must be extensive communication (i.e. data exchange) between the DU and the operational systems in the environment of the DU.

There are many different ways to implement a communication infrastructure and those often depend heavily on the actual system and platform. Suitable concepts or techniques are for example Named Pipes, Shared Memory, Inter Process Communication, Web Services (SOA) or Blackboards. In particular the Blackboard approach (e.g. Timm, Scholz, Herzog, et al., 2006) fits well in MAS systems. Figure 2.3 illustrates a basic Blackboard in an agent system. The blackboard can be some sort of data store (e.g. database system). Each agent is allowed to post on the blackboard and read the post. If a message is of relevance for an agent, the agent can consume the message, or simply ignore it if it is not relevant.

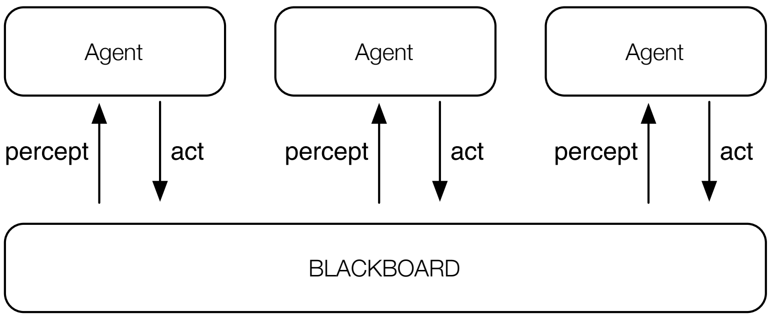

Figure 4.8 Blackboard communication - adapted from Timm et al. (2006, p. 39)

4.3.3.2 Configuration Engine (CE)

Central to the MAEBI concept is the distributed architecture that allows the localised decision making focus. However, MAEBI is not just a collection of individual systems that replaces a centralised system. MAEBI focuses on localised decision making and some decisions cannot be made on a local level. For example, tasks like Tax, TQM or corporate strategy decisions remain at a Headquarter (HQ) level and will be decided globally. The Configuration Engine's (CE) task is to manage the entire system from a centralised view. It monitors central systems, for example a DB and applies changes to the structure of the agent system. The

Configuration Engine, which in turn is also an agent, is a helper in the system where DUs cannot take required actions. For example, the CE executes the creation of a DU, as a DU cannot create itself. Once a DU is created it accesses suitable DBs to acquire data and information that is necessary to adjust it to the environment and structure (hierarchy). After receiving the start up parameters the DU acts on its own and performs tasks according the design objectives. The internal or local DB allows storing data that is of relevance for the particular DU and the current context.

4.4 An illustrative case study: Mr. Chicken

Mr Chicken is a fictitious fast-food chain that rivals the traditional fast-food chains by providing healthy yet "fast" food. The business is organised the same way as other fast food chains. Customers quickly appreciated Mr Chicken's food and the chain experienced significant growth and currently owns almost 200 restaurants across the country. All restaurants are owned by the HQ, which defines the chains overall strategy. The management of the chain is aware that the one-size-fits-all-approach is out-dated and implemented a "job-enrichment" program that gave store managers some freedom to adjust decisions like order dates, quantities and pre-production (e.g. best selling lunch burger). The program was not successful and AutoChicken was implemented, a MAEBI based system to optimise chain performance by optimising individual restaurants.

First the CE accessed the store and product databases to create and initiate the DUs that represent the individual restaurants in the chain. After the CE created the Restaurant DUs and the DUs are "alive", they begin to connect to the data sources that represent their respective environments. As the implementation aims to improve order management, the DU gathers data that describes local sales. By using its communication capabilities it connects to the (local) POS system, the (local)

inventory system and the product descriptions (recipes) from the HQ database, combines those and stores the data in its DB/DW module for reference and analysis. This data acquisition process is not a once-off step but data that changes is updated (for example recipes) or appended (for example POS data).
When data is available for further processing the DU uses its DM/KD module to analyse different aspects of its environment and "learns" or "adapts" to its (the environment) characteristics. Relevant insights are, for example, when which product sells best, product price response functions, ingredients consumption (production) and customer orders (when, how often, order size, product mix etc.).

Before the system is made operational (can make decisions) each DU goes through a learning period. In this period the DU is active but cannot implement decisions. This means that the DU analyses the data and computes a decision but this decision is not implemented (the business process is not changed by the system), however the results of the DU are stored and compared to the actual values (of the business process) to ensure that models work. Statistical error indicators and statistical confidence of the results can determine when to go operational (to be able to implement decisions). In the case that not enough data/data points are available, the system may initiate testing routines. For example, if the system cannot define a price response function for a product, it may alter the price to see how demand is changing.

In case of Mr. Chicken the insight gained into the restaurant specific characteristics allowed it to identify even minor differences. For example, previously the HQ used 12pm as start of the lunch period, and required stores to pre-cook lunch boxes. AutoChicken for example identified that Restaurant X "lunch-time" was at 13:20pm – as a school close by finishes at 13:15pm. In Restaurant Y the lunch peaks at 11:30am, as many tradesmen start work early and want lunch earlier.

Restaurant R has a number of competitors close by, however a construction site hindered customers visits to the competitor's restaurants. Store demand increased significantly and inventory of the popular burgers decreased rapidly. The DU tried to order more supplies but this will take a few days. To avoid an out of stock situation the DU alters the pricing and makes less popular burgers cheaper and the popular ones more expensive to "guide" demand. This allows the store to profit from the situation and customers "think" they are in control.

Managers at HQ still have relevant aggregated data available that is required for administrative task like accounting, tax or finance. Because high granularity data (eg. item level POS) is already processed at store level less data has to be transferred between restaurants and HQ, which reduces communication and computing expenses. In addition the central marketing department has more detailed information on their customers and how they differ. These insight can be used for more targeted marketing.

One aspect of the AutoChicken system is to give stores some freedom in ordering their supplies individually. General agreements with suppliers remain a HQ matter. However, buyers now have data that better reflects actual (chain) demand and can negotiate more effectively with suppliers.

4.5 Distinction to similar areas

There are technologies, concepts and architectures that may seem to have similar attributes, characteristics or functionality as the proposed MAEBI architecture. This section compares MAEBI to technologies that comes closest to the proposed concept and highlights key differences.

4.5.1 MAEBI and MAS

MAS promise to be more flexible than conventional software development and design methods, primarily because agents are, to some extent, autonomous and flexible. Table 4.1 maps the individual agent attributes identified by Padgham and Winikoff (2005) to the MAEBI concept and describes how they affect the MAEBI concept.

Agent/MAS Characteristics	MAEBI Characteristics
Situated	Is embedded in a context (e.g. store level, organisational level)
Autonomous	Each DU can decide on its own
Reactive	Can sense (gather) data from environment and take actions
Proactive	Learns from past and can proactively alter environment
Robust/flexible	'Catches problems' (e.g. rules can capture DM problems)
Social	Communication facilities
Rational	Clear KPIs / Goals

Table 4.1 Agent Characteristics mapped on MAEBI (adapted from Padgham & Winikoff, 2005)

4.5.2 MAEBI vs. SOA

Web Services and Service Oriented Architecture (SOA) are software design concepts that focus on re-usable services rather than on complete applications (see Chapter 2). SOA has overlapping aims with MAEBI. MAEBI uses the de-coupled (or loosely coupled) nature of Agents in MAS systems to distribute decision making capabilities throughout the organisation. Even though one can argue that a DU is a service similar to a service in SOA, the significant difference is that a DU is an autonomous software implementation and not just an (distributed) interface to some central system. Georgeff (2009) argues that in an increasing complex (business) world, where software (services) engage in inter-organisational communication, it becomes too complex to orchestrate such services. He summarises SOA as follows "In short, most of the promised benefits of a loosely coupled SOA get lost in the tight coupling of business processes and tightly linked control and data flows." (Georgeff, 2009, p. 394). As such SOA and MAEBI are not competing concepts. The communication module in a MAEBI DU may utilise provided SOA services to connect to business processes or other external databases.

4.5.3 MAEBI vs. 'traditional' BI

BI traditionally is a relatively rigid process where data is aggregated from various operational sources and stored in a data warehouse where it is accessible for later use. Despite the improvements in BI it is usually still a centralised system that aims to support mainly strategic decisions. The presented MAEBI framework builds on BI and develops the concept with focus on localised decision making. MAEBI adopts the functional modules of BI (DW, ETL, DM/KD) and encapsulates those into agents to increase flexibility. In addition to the "traditional" modules, MAEBI also includes decision execution functionality to implement a decision into the process. The local focus of the system however does not mean that centralised aspects of BI are obsolete. Certain tasks and responsibilities, for example accounting, tax or strategy development will remain at HQ level.

4.5.4 MAEBI vs. Distributed Data Mining (DDM)

One of the objectives of MAEBI is the focus on localised data analysis to improve decision making for the respective environments (e.g. customers of a store instead of “customers of a chain”). Each DU is equipped with a DM/KD module that implements one or more data mining method(s) that are suitable for the task.

Data mining in a non-centralised fashion is often referred to as Distributed Data Mining (DDM). DDM is a concept where the model building (i.e. training) is split over different databases and computer systems and later combined to one final model. Figure 4.9 shows a “traditional” data mining/data warehouse system; the right side of the illustrations shows a DDM system. The workflow in the centralised system is, that all data is copied to a central DW and then fed into a data mining tool to train a mining model or gain some result. In a DDM system every node generates a local model, based on local data. All local models are then combined to produce a final model.

Similar to DDM, MAEBI performs analysis of local data and generates/trains local models, however DDM is a concept for mining large/distributed databases whereas MAEBI is an integrated concept for local decision making. In a DDM system the final model is of interest (Figure 4.9 right), a DU in a MAEBI based system utilises a local model for decision making. Generally, MAEBI is not to be understood as a data mining tool, DM is one of the techniques that is used to deliver decision making capabilities throughout an organisation (or business).

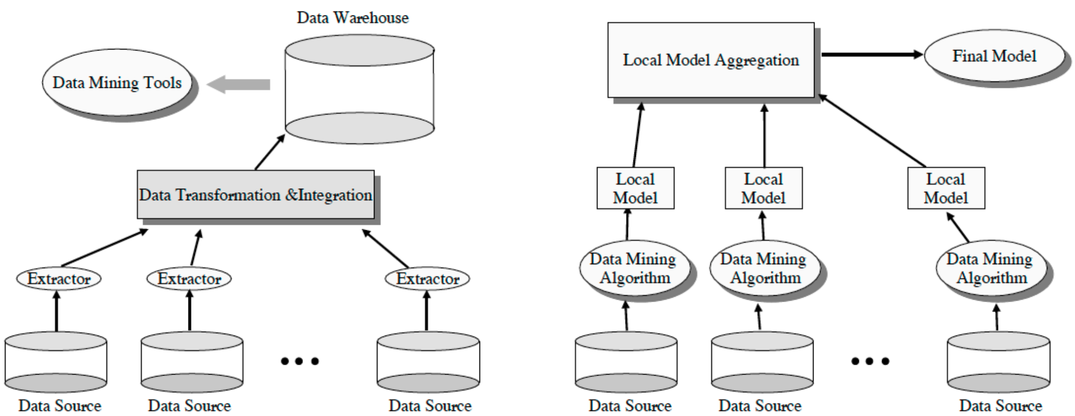

Figure 4.9 Data Mining vs Distributed Data Mining (adopted from Park and Kargupta, 2002)

4.6 Summary

The chapter first transformed some of the issues identified in the literature review into design objectives for a solution. Five different design objectives were identified to advance the current BI concept. Agent and Multi Agent technology was proposed as an option to implement the design objectives. These activities represent activity 2 in Peffers DSR process.

Section 3 describes the design of the MAEBI concept and the two components, the Decision Unit and the Configuration Engine. This design stage represents activity 3 in Peffers DSR process. The purpose of the individual components and their modules was explained and some comments about possible implementations options were made.

To better communicate the "idea" behind the approach, a synthetic case study in the context of a food chain was presented. Finally, core aspects of the MAEBI concept were compared to similar (existing) technologies to show how they differ.

Chapter 5 - Design Evaluation (pricing MAEBI)

5.1 Introduction

The previous chapter described the MAEBI concept based on 5 design objectives that all aim to better align BI to business requirements that were identified in literature. Central to the MAEBI concept is the decision unit (DU), that encapsulates decision making capability and can be placed into the problem domain. Following the research process outlined in chapter 3, the next steps are to demonstrate (Peffers et al. (2008) Activity 4) and evaluate (Peffers et al. (2008) Activity 5) the proposed artefact.

According to Peffers et al. (2008), to appropriately demonstrate an artefact, it has to be applied to “one or more instances of the problem”. This can be done for example as simulation or experimentation. In either case the requirements are defined by the business context (Hevner, et al., 2004). The business context chosen for the demonstration system is item level pricing in retail chains, as it allows showcasing different capabilities of the proposed MAEBI concept.

BI systems and DSS in general can be applied in many different ways and contexts that make demonstration and evaluation difficult. The main goal of the protoptype is to show the feasability of the MAEBI concept in a complex environment. In this relation Vaishnavi et al. point out (2007, p. 21) “The implementation itself can be very pedestrian and need not involve novelty beyond the state-of-practice for the given artifact; the novelty is primarily in the design, not the construction of the artifact.”

Figure 5.1 illustrates the proposed testbed to demonstrate the MAEBI concept in the pricing context. The testbed comprised of three parts, the pMAEBI system, a custom retail simulation and a “centralised” system for comparison. The design and implementation choices are described in this chapter followed by details and results of the simulation.

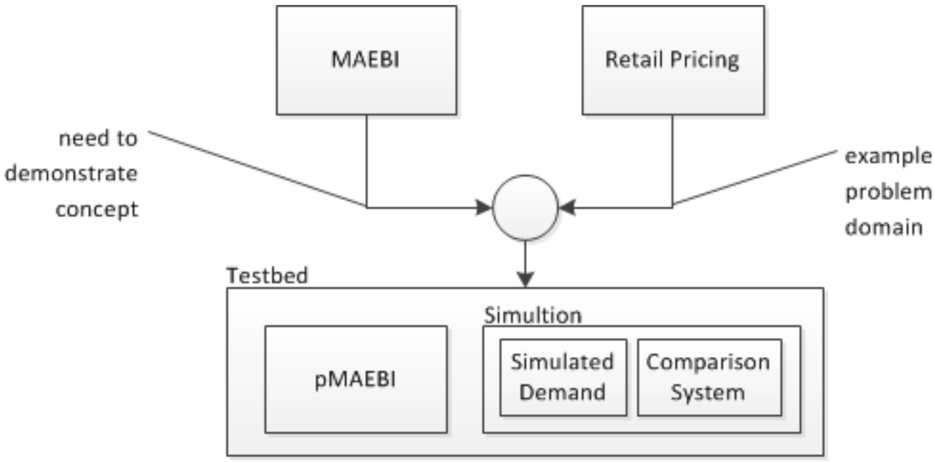

Figure 5.1 Chapter 5 Overview

5.2 Problem Domain - Retail Pricing

The outcome of a design science research (DSR) should be the solution, or improvement, to a relevant business problem and that solution has to be demonstrated (Hevner, et al., 2004). In contrast to an (standard) implementation, DSR outcomes have to provide a contribution to knowledge. MAEBI’s design objectives particularly aim at making decisions in complex environments, such as retail pricing. Zentes et al. (2007, p. 191) present different pricing methods and argue that pricing is complex due to the “… intense interdependence of influence factors …”. The authors further argue that this complexity led in the past to simple pricing rules (e.g. cost + X% margin) and that it is necessary to develop new technologies to support the pricing process to incorporate factors like cost, competition and customers (i.e. demand). The complexity of pricing increases in retail chains (or multi store retailers) as demand can be different at different time in the day in different stores. Trivedi (2011) analyses store and category data and identified “distinct location patterns”. Rigby (2006) argues from a management perspective that localisation is the ‘revolution’ in consumer markets and that one-size-fits-all does not address customer’s needs anymore. Marn and Rosiello (1992) paper found that a 1% improvement in price can lead to an average of 11% in improved profits. Their research had significant impact on pricing research and is the motivation behind many research efforts in the field. How the MAEBI concept is applied to the problem domain is illustrated in Figure 5.2.

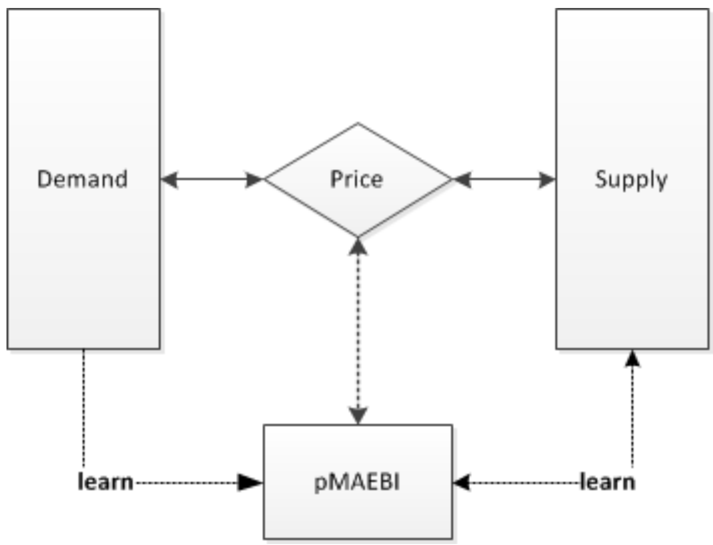

Figure 5.2 Demand - Price - Supply

Supply and Demand are “connected” through the price, the pMAEBI system learns from both sides by analysing demand data and product information (e.g. cost) and based on this insight a price for a product is implemented.

5.3 Testbed System

In section 3.4.5 a testbed was proposed that is consistent with design science research methodology (Hevner, et al., 2004; Peffers, et al., 2008) and agent literature (Theodoropoulos, et al., 2009). Placing this into context of the problem domain (compare Figure 5.2) requires that the demand and supply ”blocks” have to be simulated to be able to analyse the proposed system in the environment it is designed for. To conceptually follow the idea of analysing the designed artefact in its environment, the design guideline for the simulation is a software – in – the loop architecture, illustrated in Figure 5.3.

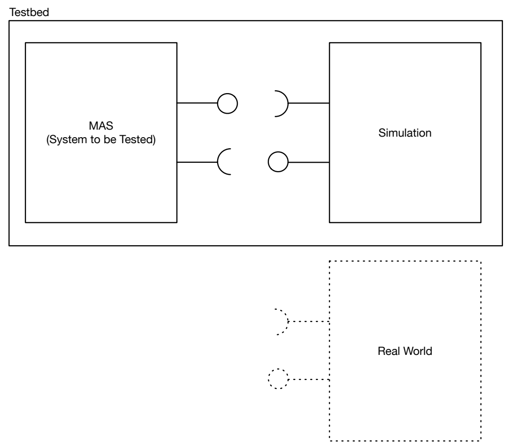

Figure 5.3 Software-In-The-Loop Testing

The idea behind this architecture is to design the testbed as distinct parts, where both parts (the simulation and the system to be tested) have the same “interface” as they would have in the real world. This means that the simulation (in this testbed) exposes sales data (through a database); the pMAEBI system consumes that data (through a database interface) and makes decisions based on that data. These decisions (i.e. a new price for product) are fed back into the simulation database (the loop). The simulation uses that new data “the next time” (whenever the data is required next in the simulation).

Figure 5.4 depicts the building blocks of the testbed system, which is implemented on various Microsoft technologies and runs on the Windows Operating System. The components are developed using the .NET and/or the Axum framework. MS SQL Server and Analysis Services (SSAS) are off the shelf components used as database and data mining system respectively. The simulation component simulates the retail chain (demand) and integrates with the centralised system “HQ” (control system) and the pMAEBI demonstration system.

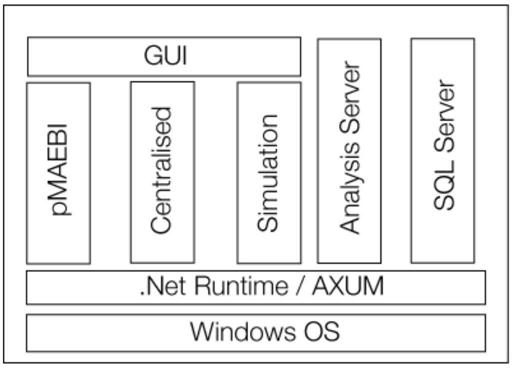

Figure 5.4 Testbed Architecture

5.4 Tools and Technologies

5.4.1 MS SQL Server

The implementation of the testbed uses some off the shelf systems, in particular Microsoft SQL Server 2008[7]. MS SQL Server is a suite of servers, that include a relational database server (SQL Server), Analysis Services (SSAS) that allow multi dimensional data analysis (OLAP) as well as data mining (DM). Other components are Integration Services and Reporting Services, which weren’t used in the testbed.
SQL Server was chosen because of its versatility and the ease of integration between the components and the development environment and previous knowledge and

[7] http://www.microsoft.com/sqlserver/en/us/default.aspx

experience of the author. The choice does not imply any judgment on the quality of the software.

5.4.2 Axum

Agent and Multi Agent technology was identified as a means to realise and implement a MAEBI based system and to address the design objectives. To develop agent based system special development languages or systems are required that differ in terms of metaphor from "traditional" OOP languages (e.g. classes vs. agents).

The agent language used to implement the pMAEBI prototype is Axum. (Gustafsson, 2009a, 2009b). Unlike most agent system that are based on Java and are result of academic research, Microsoft developed Axum syntactically very close to C# with influence from languages like Scala and Erlang.

Axum is an incubator language and available literature is limited to a few documents that the developers released (Gustafsson, 2009a, 2009b; Microsoft) on the project's website. None of the documents reached version 1.0. Some additional "experimentation" was required during the development process of the pMAEBI development to get satisfying results. Experimentation in this context refers to the creation of "little test" programs to get accustomed to how certain constructs behave in Axum.

Section 2.3 presented various aspects of agent languages and multi agent systems, how fragmented the field is and that there is no 'universal' solution. Axum, while not complete, does provide the required functionality to develop agent systems and, as it is based on C#/. Net, provides an approach that is very close to "general purpose" languages. In other words Axum allows practical implementation of an agent system without adding a layer of complexity (e. g. runtime environment) or being limited to a specific design methodology.

5.4.2.1 Agent & Channel

In Axum a channel is the construct that allows communication between agents and contains one or more ports. The concept is similar to the class / interface combination in object oriented programming. In other words, a channel (in Axum) is like a "communication contract" between agents and the ports define the type of data that can be used in a particular channel. An agent is the organization unit that contains the program code to perform actions on data. Every agent must implement a channel. Instead of instantiating an agent directly, the channel that is implemented by that agent is instantiated.

Section 5.5 and subsections cover implementation details and will introduce the design of the Agents, CE and DU.

5.4.2.2 Domain

In agent systems message passing is a core concept to share data between agents. To pass messages between agents, data must either be copied or be immutable to ensure that the data is transferred correctly (e.g. to avoid any race conditions). Copying data, as a means of communication, is often not an efficient method (e.g. memory usage) and not all data is immutable. In Axum a domain is an isolation unit that has similarities to a class in object oriented programming. A domain allows sharing data between agents that are "in" the domain (using a reader/writer approach). An Axum domain can contain agent declarations, fields and methods and isolates those from other domains.

There are two domain types defined in pMAEBI, the program domain, which exists only once, and the store domain. The program domain, which is the default domain of an Axum application, acts as the entry point after the (Axum) application is started. It contains code to receive and process possible start-up parameters (command line arguments). In the pMAEBI prototype the Axum program is started (using the GUI application) with a simulation ID (simID) as parameter. This simID allows access to the relevant information in the simulation database to instantiate the respective stores and products that are represented by the pMAEBI system.
The second domain type, the store domain, represents a retail store and groups the DUs that "belong" to a store. Thus, the domain is conceptually the local environment for the DU agents.

5.5 Pricing MAEBI (pMAEBI)

5.5.1 Configuration Engine (CE) Implementation

The purpose of the CE is to manage the overall system and take action where local perspectives might be not sufficient, for example the creation of DUs. The CE in the pMAEBI system is limited in terms of features and responsibilities as features in an experimental system are hardcoded. Main responsibility of the CE on the pMAEBI system is the "creation" of the DU system, that is the generation (instantiation) of DU agents based on the information about the chain/stores from the database system.

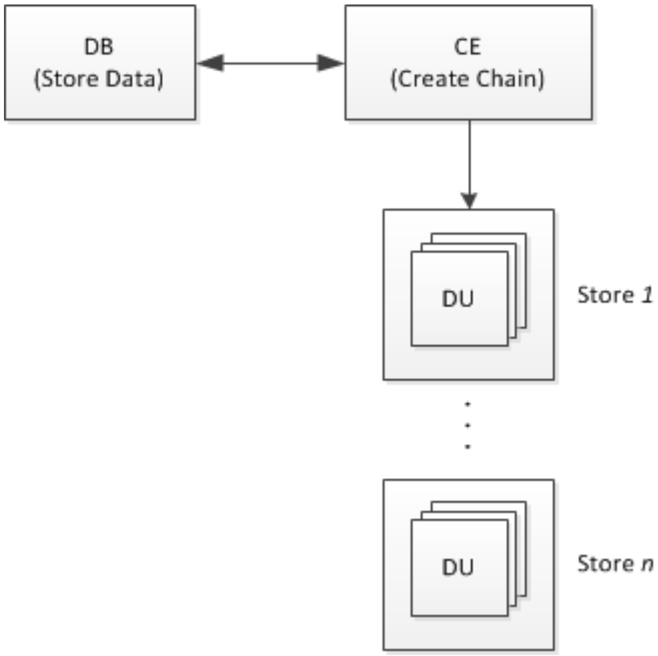

Figure 5.5 CE Workflow

Figure 5.5 shows the implemented workflow of the CE in the prototype system. It connects to the database and reads information about the individual stores and the store's product offerings (Store IDs, Product IDs). Based on this data the CE creates the respective DUs and places them into the right domain (store).

5.5.2 Decision Unit (DU) Implementation

The second building block in MAEBI is the decision unit (DU) (Figure 5.6) and the core of the actual system. A DU is the construct that bundles the functionality that is required to provide decision making capabilities. Five different features make up the DU, Communication, Database, Data Mining, Learning and Decision Execution. Each module can be implemented in different ways using different technologies and may vary in significance.

The following sections describe the implementation of the modules in the pMAEBI prototype.

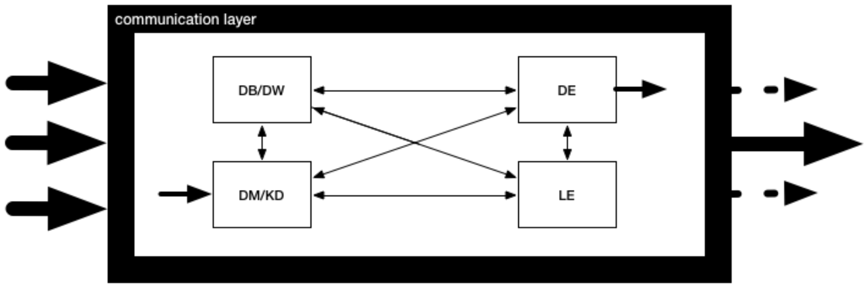

Figure 5.6 Decision Unit (DU)

5.5.2.1 Communication

A DU has to communicate with its environment (other DUs and/or other systems) to access data and information and to transfer and implement results. As systems and implementations vary, there are different technologies and methods applicable. The method used for the pMAEBI system is a very basic blackboard approach that allows communication between components using a database as the "board". All components that are included in the black board system were assigned a GUID during the initialisation phase, which is used to send and receive messages.

Technically, pMAEBI is based on .NET / Axum and thus uses the ADO.NET framework (ActiveX Data Object for .NET) to connect to SQL Server.

Two additional APIs were used to allow communication with SSAS, SQL Server Management Objects (SMO) and ADOMD.NET. The former allows administrative access (e.g. create, alter or delete objects in SSAS) to SSAS and latter allows submission of predictive queries (see 5.5.2.3).

5.5.2.2 DB/DW

A key component of most DSS and BI systems is a database or data warehouse to store data and information for the purpose of later analysis. Also systems need storage to save configuration and other temporary data.

There are different ways to implement (structured) data storage, ranging from a array of variables, through to constructs like DataSets[8] or relational databases. The DU in the pMAEBI system uses SQL Server (relational database).

Depending on the implementation context a DU must be capable of storing different types of data, foremost however data that describes the problem. In the pMAEBI system that is the product sales data.

5.5.2.3 DM/KD

The DM/KD capabilities of a DU are used to analyse data from the problem domain and gain actionable information to improve or solve a particular problem. Depending on the problem domains and problem itself, different implementations can be imagined. Advances in data mining and knowledge discovery have led to a choice of algorithms and methods that can be employed. Open source and commercial "out of the box" systems are also available, like SSAS the system that was used to realise the DM capabilities in the prototype system (e.g. Janus and Fouche, 2009). SSAS provides several mining methods, however only the Artificial Neural Network method was used in the pMAEBI system to analyse sales data and determine the price for a product. For practical reasons (e.g. system overhead) only one instance of SSAS was implemented. However to separate the individual DUs in

[8] http://msdn.microsoft.com/en-us/library/system.data.dataset(VS.71).aspx

SSAS, each DU is represented with its own mining model instance within the SSAS instance. This means each DU had full control over its own mining model.

Figure 5.7 shows the process flow within the DU around the DM module. The data is gathered from the database (here product sales data) and used as input for the data mining model, which is a distinct connection to a model instance in SSAS. The prediction (i.e. the result) of the mining process is then transferred to the DE module of the DU.

The pricing process is described in depth in section 5.6.5 of this chapter.

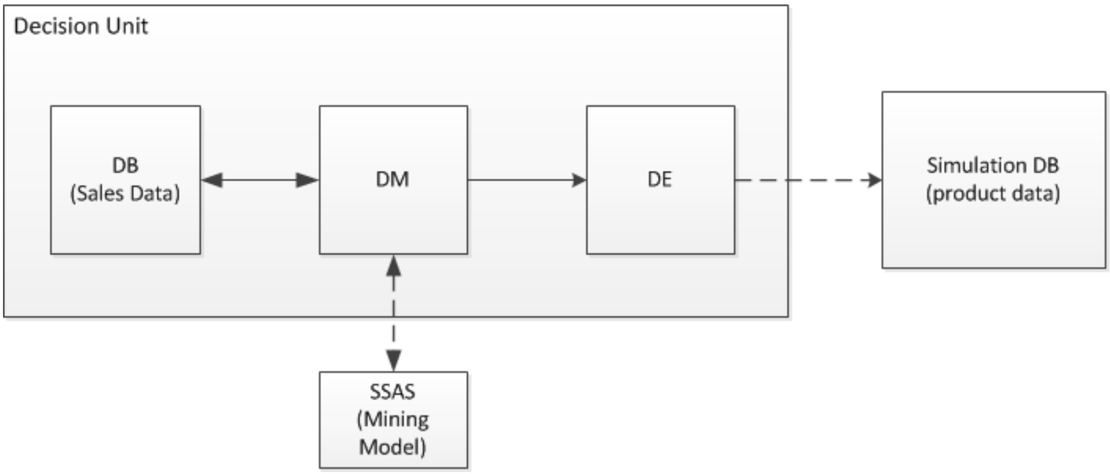

Figure 5.7 DM/KD Workflow

5.5.2.4 Learning

One of the MAEBI objectives is to adjust to the environment, “learn” from that environment and integrate this information (or knowledge) into the decision making process. In context of this prototype the learning is related to the pricing algorithm implementation. The environment refers to the consumer, specifically to the buying behaviour of the customers in the respective stores.

The learning capabilities in the pMAEBI system are integrated with the DM/KD module and the pricing method described in 5.5.2.3 and 5.6.6.

5.5.2.5 Decision Execution

In contrast to "traditional" BI systems, MAEBI based systems are capable of implementing a decision into operational systems. Besides having access to the respective operational systems and the required access rights, there must be functionality to check if the output of the DM/KM module is a viable decision. This means that the result of the data mining process might not be a practical or legal solution. For example, based on the available data the optimal price of product might be $1.03, however there are agreements with the manufacturer that the retailer charges at least $1.49. The goal of the MAEBI is to consider and leverage local data and knowledge, but this does not mean ignoring procedures and requirements of the corporation (global view).

Within the pMAEBI prototype system, the DE module implements two functions, 1) connectivity to the operational system (here the simulation system) and 2) enforce one rule, that is the price of a product cannot be less than the cost of the product.

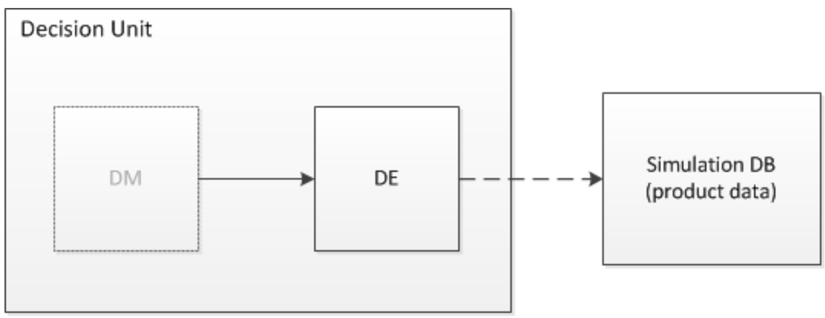

Figure 5.8 DE Module

$$P_{ij} = \left\{ \begin{array}{c} p_{ij}, p_{ij} \geq c_{ij} \\ c_i, p_{ij} < c_{ij} \end{array} \right\}$$ Equation 5.1

with

P_{ij} Actual New Price for Product *i* in Store *j*

p_{ij} Suggested New Price for Product *i* in Store *j* from DM module

c_{ij} Cost of Product *i* in Store *j*

Figure 5.8 illustrates the workflow of the module. The input of the DE module is the "proposed" sales price of an item that the DM/KM module determined based on the input data. If required, the DE adjusts that price based on the implemented rule(s). The DE module implements only one rule in the pMAEBI system. Specifically it ensures that the new price of a product is at least equal to the cost of the product (Equation 5.1). The price is then updated in the product database (simulation).

5.6 Simulation Design

5.6.1 Simulation Objectives / Outcome

The purpose of the simulation in the testbed is twofold, 1) to simulate demand and 2) act as the comparison system (centralised BI). There are three different objects defined in the simulation, Customer, Store and Product. Central to the simulation is the customer object that represents a single customer and its characteristics as the "source of demand". The type of data that is generated in this process is commonly known as point of sales data (POS Data), the type of data that most know from a grocery store receipt (Table 5.1).

POS DATA	
Transaction ID	GUID for the Transaction
Line Item(s)	[Product ID, Quantity, Unit Price, Total]
Store ID	ID of the Store
Total	Total $ Amount
Time Stamp	Simulation Time Stamp

Table 5.1 POS Data

In contrast to simple data generators, the focus is not just to generate data and analyse how the system under study (pMAEBI) processes the data, but to analyse the interaction between the system and its environment – the "consequence" of decisions made by the pMAEBI system.

Simulating demand in terms of the buying behaviour of individual customers is difficult, however this granularity of input data is required for the pMAEBI system and the data mining activities involved. The objectives for the simulation system are 1) to design it conceptually correctly (i.e. that the customer goes through a decision process before buying a product and that depends on a set of decision variables) without drifting too far in to marketing and psychology, 2) that it delivers the right granularity of data and 3) that it can react to the output of the pMAEBI system.

The individual building blocks of the simulation, timing and workflow are explained in this next section.

5.6.2 Assumptions

The simulation aims at being conceptually correct but does not attempt to represent the entire complexity of the retail / customer interaction. A few assumptions and limitations underlie the simulation. These assumptions may not necessarily be realistic, however they provide a much simplified environment to assess the design of the MAEBI concept - the focus of this research.

- Change in Quantity sold has no impact on Cost – there are no economies of scales effects
- There is an endless supply of product - No out of stock situations arise
- Price $\geq$ Cost i.e. no loss leader situations
- Cost ≥ 0.25 (the minimum prices chosen for this simulation)
- Products are independent - there are no substitutes available
- Customers are perfectly rational and buy based on the value function of product
- All variables (such as customer and product characteristics) are discrete in the set $\{0, 0.25, 0.5, 0.75, 1\}$

Throughout the simulation a "general attributes concept" is used that describes characteristics of a simulation object on a discrete scale. This approach is derived from Neri (2007) and Jager (2007). The simulation here is used to show a proof of concept implementation of the MAEBI concept, it is not to be understood as a simulation that captures all of the complexity involved in consumer behaviour. Yet, to have a simplified model that is 'conceptually' correct, each customer and product object contains three general attributes. These attributes could represent factors like, quality, availability, taste, experience, image etc. Each of the factors can take a discrete value from the set $\{0, 0.25, 0.5, 0.75, 1\}$. For example, assuming one of the attributes is quality and is assigned a value of 0, this means that this is a low quality product, whereas a value of 1 would describe a high quality product. As the price is of particular interest in the simulation, it is explicitly modelled in the product object, based on the set {0.25, 0.5, 0.75, 1}.

This approach allows us to capture some of the complexity involved in consumer behaviour and present a suitable testbed for the MABEI environment.

5.6.3 Simulation Process

Figure 5.9 shows the simulation process. A simulation run is started, the "retail chain" is initiated and a timer object is created. After all simulation objects report to be "runnable" (i.e. completed the initiation phase) the timer sets the simulation status to run which in turn activates all other objects. During each simulation step, all customers of all stores "go shopping" and either purchase a product or don't based on the product characteristics and the personal preferences (value function).

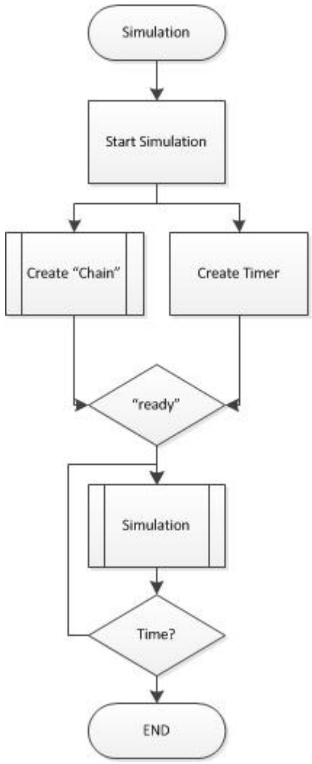

Figure 5.9 Simulation Process

5.6.4 Simulation Objects

5.6.4.1 Store

A store object represents a store instance in the simulation and groups customers and products together. A store object contains a list of products that are available and two attributes (Table 5.2). During the initialisation phase of the simulation, a store is always created twice (as a pair) with the same customers (same characteristics but different customer objects) and same products (same characteristics but different product objects). This allows to analyse the "same" store using the "traditional" centralised approach and the new pMAEBI decentralised system. The HQStore attribute distinguishes the two "versions".

Attribute	Description
Store ID	ID of Store
HQStore	HQStore = 1 -> Control System HQStore = 0 -> pMAEBI System
Products	List of Products that are sold in a particular store

Table 5.2 Store Object Attributes

5.6.4.2 Product

A product in the simulation is represented by a product object and defined by five different attributes (Table 5.3). Each product object represents a product/store combination.

Attribute	Values	Description
Attribute 1	$\in \{0,0.25,0.5,0.75,1\}$	"General Purpose" attributes that quantitatively describe product characteristics
Attribute 2	$\in \{0,0.25,0.5,0.75,1\}$	
Attribute 3	$\in \{0,0.25,0.5,0.75,1\}$	
Cost	$\in \{0.25,0.5,0.75,1\}$	Cost of the Product (total & fixed)
Price	$\in \{0.25,0.5,0.75,1\}$ $P \geq C$	Current Price of the Product in Store *S*. Price must be equal or greater than the cost, no "Loss Leaders"

Table 5.3 Product Object Attributes

Attributes 1 – 3 are “general purpose” attributes that quantitatively express the product’s characteristics and match the customer attributes (see 5.6.4.3). This allows the calculation of a perceived value or utility of a product to an individual customer. The calculation is adopted from Jager (2007, p. 870) (Equation 5.2).

$$U_{ij} = \frac{\sum_1^n (\beta_{in} * U_{ijn})}{n}$$ Equation 5.2

with

U_{ij} Utility for consumer i of product j, ranging from 0 to 1

β_{in} Weighing of attribute n for consumer i

U_{ijn} Utility for consumer i of product j for attribute n

n Number of Attributes

(The weights β_{in} of the three attributes are assumed to be equal for this simulation)

Price and cost are characteristics that are of special interest for this simulation and are explicitly implemented. The price of a product is what the pMAEBI system should adjust. Product price in combination with the cost of the product allows the system to calculate profit/loss and quantify the objective of the system. The cost of a product is total and fixed, i.e. the costs of a product for the retailer do not change over time and all cost components (e.g. fixed, variable, discounts, shipping etc.) are reflected in the product price.

5.6.4.3 Customer

Simulating demand is challenging, in particular customer buying behaviour, as motivation or triggers to buy a product can be complex. Marketing literature is of limited help as models are usually qualitative and cannot be directly transferred to code. The focus of the customer design is to have a model that explicitly formulates a decision function that takes the price of the product into consideration.

<table>
<tr><th>Attribute</th><th>Description</th></tr>
<tr><td>Customer ID</td><td>ID of Customer</td></tr>
<tr><td>Attribute 1</td><td rowspan="3">“General Purpose” attributes that quantitative describe customer characteristics/ preferences.</td></tr>
<tr><td>Attribute 2</td></tr>
<tr><td>Attribute 3</td></tr>
<tr><td>Preferred Store</td><td>The store where the customer shops</td></tr>
<tr><td>Budget</td><td>Budget Indicator (1 = high, 0=low)</td></tr>
</table>

Table 5.4 Customer Object Attributes

A customer is described by six attributes, an ID that identifies the customer, the preferred store (a customer only shops in 1 store), three “general purpose” attributes that express the customer quantitatively (see 5.6.2) and an indicator of the available budget. The values of these attributes are from the set $\{0,0.25,0.5,0.75,1\}$.

The customer decides, based on the personal value of a product and the available budget, whether or not she will buy the product. At each shopping round this value is calculated for every customer/product combination and evaluated as to whether the value is enough to trigger a purchase. The product value function (Equation 5.3) is adopted from (Jager, 2007, p. 871).

$$V_{ij} = U_{ij} * B_i * (1 - P_j)$$ Equation 5.3

with

V_{ij} Value for money of product j for consumer i

U_{ij} Utility for consumer i for product j

P_j Price for Product j

B_i Budget of consumer i

The value function brings the utility of the product (Equation 5.2) in relation to the price and the budget of the customer. To eventually trigger the buying decision the value of a product for a customer V_{ij} must be greater than (1-B_i). This links the budget of a customer, the utility and the price (value) of a product.

The designed and implemented customer process is shown in Figure 5.10. A customer object gets its initiation data from the database and waits for the simulation to start. The simulation timer then triggers new ‘shopping trips’ for the customer objects. Each customer then loops through each product that is offered in the “Preferred Store” and decides, based on the value of the product, whether to purchase that product.

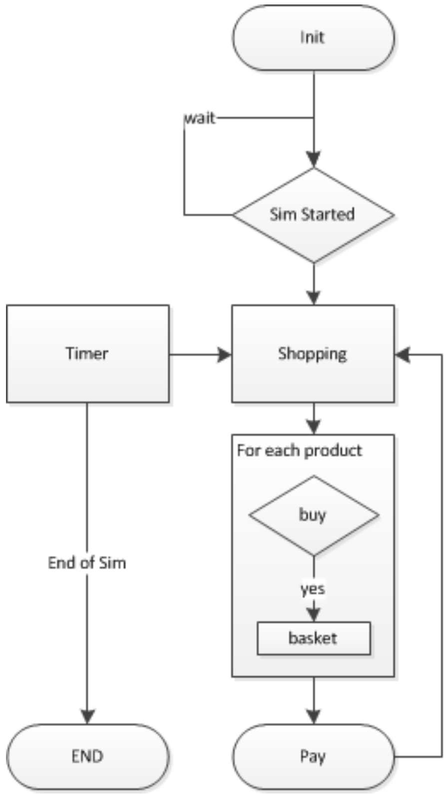

Figure 5.10 - Customer Workflow

5.6.5 Timing

Simulation timing is handled by a separate thread, which follows the process depicted in Figure 5.11. Time is recorded in week slices; there are 3 slices a day, which equates to 21 slices a week. This makes it easier to identify parts of the week for the data mining algorithm.

The process of the simulation timer is very basic. After the user starts the simulation, the timer thread waits for all other objects to be in a "runnable state", and then sets the simulation status to "run". During each iteration it posts the current simulation time to the "blackboard" and eventually sets the simulation state to "end" when the timer expires.

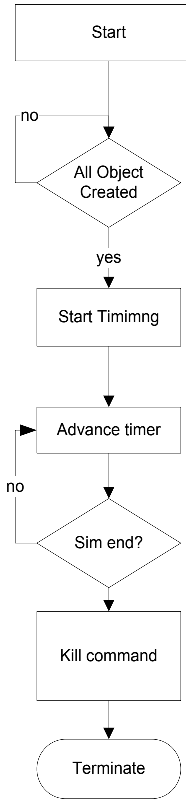

Figure 5.11 – Timer

5.6.6 Pricing Process

The pMAEBI system applies the MAEBI concept to the retail context. Specifically the pMAEBI system makes item level pricing decisions according to store level demand characteristics. This pricing method was implemented using the learning and DM/KD module of the DU by utilising the ANN mining model from SSAS. Its goal is to determine the price for a product/store combination that reflects the demand characteristics of that particular store.

The pMAEBI system uses a simple approach to determine the price for individual products. As noted in the literature survey, there are a number of challenges involved in pricing and there are a multitude of qualitative and quantitative approaches documented. The pricing algorithm used here should be conceptually correct to illustrate the function of the system. This algorithm does not claim to be

either new or optimal. The objective of the system is to improve (i.e. increase) profits.
For demonstration purposes the pricing method should be based on a techniques that is commonly found in BI systems, ANN is such a technique. ANNs are a versatile nonlinear technique that can be used for classification and regression and can learn from data (Alon, Qi, and Sadowski, 2001; Khan, et al., 2008).

Every product is assigned a price (i.e. recommended retail price). To allow the system to collect sales data at various price points (to learn about price/demand), the first week of the simulation is used for "Price Testing" (Dolgui and Proth, 2010). During this price testing period the price of a product is not calculated but a random price is selected from the discrete set (see assumptions) and assigned to the product as an initial price.

To predict future demand, respective price/sales data of a product has to be analysed in form of a time series. The mining model used in SSAS is the Microsoft Artificial Neural Network algorithm (e.g. Alon, et al., 2001). ANNs were chosen to better reflect the relationships between unit sales and varying prices over time. ANNs are used in different marketing areas. Parsons et al. (2003) stress the general versatility of ANNs and that the approach should approximate traditional statistical methods. Other methods could have been chosen, but the actual implementation of mining models in SSAS is practically the same for all supported methods. This research does not compare pricing algorithms, thus the only requirement is that the algorithm is correct. ANNs are one feasible option in this context and as it did not require additional resources to implement, this method is used. It is important however, that all systems use the same algorithm and that this influence factor is kept 'constant'.

Inputs to the mining module are:

- Time (Key)
- Week Slice
- Sales for Period
- Price for Period

Output of the model is unit demand at t+1.

Profit is calculated for every product/store combination independently based on Equation 5.4. Cost of a product is assigned during setup and constant during the simulation and can be seen as a Total Cost (Fixed + Variable).

$$R_{it} = (C_i - P_{it}) * D_{it} \qquad \text{Equation 5.4}$$

Revenue

with

R_{it} Revenue for Product i at time t

C_i Cost for Product i

P_{it} Price for Product i at time t

D_{it} Demand for Product i at time t

5.6.7 GUI

A basic user interface was designed and implemented to make the interaction with the system easier, in particular the creation of new simulation runs and the start process (Figure 5.12).

The user can use the GUI to create a new simulation run. This triggers that all the required databases and tables are generated and populated with the correct start values. In addition a new SSAS database is created for the data mining activities.

Besides the creation of a simulation run, the GUI allows the user to start and kill the simulation and the associated threads. There is an additional "manual" start mode that allows the user to monitor and intervene in the start up process. This was implemented for debugging reasons. The "normal" start method starts the simulation and it ends after the timer expires.

While the simulation is running, the GUI indicates the status of the simulation timer as a "traffic light" to have some feedback. The GUI does absorb some of the complexity of the software, however it was not designed to fully control all aspects of the system.

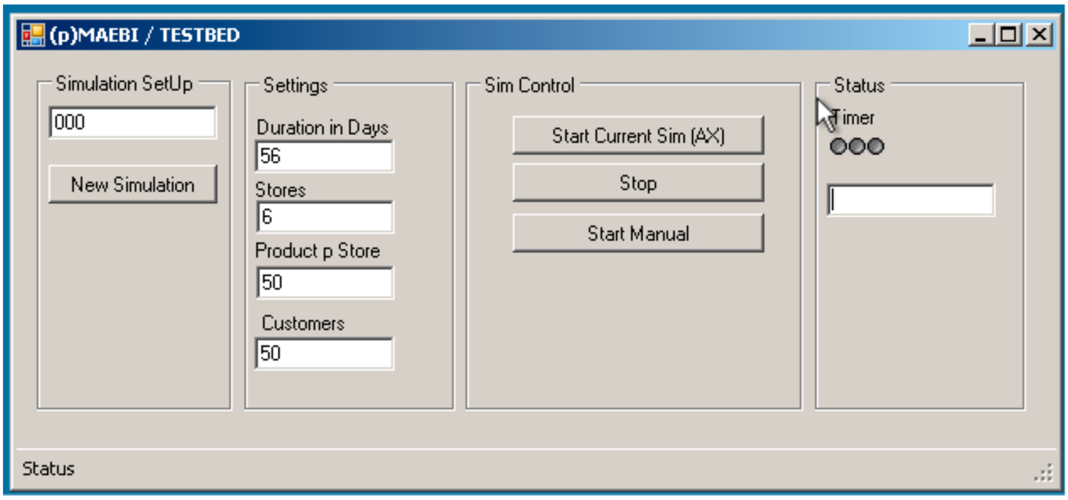

Figure 5.12 GUI Screenshot

5.7 Simulation & Analysis

This section summarises the result of the simulation performed that was described in the previous section. The simulation represents a hypothetical economy and compares a centralised / traditional BI approach with the pMAEBI system that is based on the MAEBI concepts proposed in this research. This means that, if the stores that use the pMAEBI system perform better – better means higher profit – will be understood that the new concept would be viable. In regards to the absolute pricing performance, this implementation is not a pricing system that aims to find the 'best' price.

For every simulation run the test bed software creates a new set of SQL and SSAS databases. During the initiation stage, parameters like duration, store count and product count are read from the GUI and the required data is inserted into the respective DBs. This completes the simulation setup process. The user starts the simulation timer object that controls the simulation. It will wait till other simulation objects report that they completed the initialisation process and are in a 'runnable' state. The timer then sets the simulation status to running and increments the simulation clock until the simulation is complete.

5.7.1 Simulation Runs

Each of the executed simulation runs was started with 3 different stores pairs (6 individual stores). Each store has a customer base of 50 unique shoppers. The duration of each simulation run is 8 virtual weeks.

The simulation was limited to 3 Pairs / 6 Stores because of performance reasons that were identified during the development process. This is related to the available hardware that was used and is not related to the architecture.

5.7.1.1 Products

To populate the product objects in the simulation, a set of 150 unique products is used. Table 5.5 (Table 5.7 explains the variables) shows examples of these products (Initial_Price and Current_Price are the same at the start; Current_Price is the only attribute that can change during the simulation). In the initiation phase of the simulation each (unique) product is copied twice into the simulation database with different product IDs. This is done to have "the same" product for each store of a store pair (pMAEBI store and comparison store).

Table 5.6 (Table 5.7 explains the variables) summarises the averages (of all 6 stores in a simulation) of the product price changes over three simulation runs (grouped by Simulation ID). The averages of the Start_Price are the same (the same set of the 150 initial products). The different averages of the End_Price show that the simulation reacts differently in simulation runs.

Product_ID	A1	A2	A3	Initial_Cost	Inital_Price	Current_Price
1596	1	0.5	1	0.25	0.75	0.75
1606	0.25	0.75	0.75	0.75	0.75	0.75
1652	0.5	1	0.75	0.25	1	1

Table 5.5 Products (Examples)

SIM_ID	Start_Price	End_Price	Change
667	0.76	0.743333333	-2.19%
681	0.76	0.7875	3.62%
682	0.76	0.684166667	-9.98%

Table 5.6 Products in Simulation

Column	Description
SIM_ID	ID of Simulation Run
Product_ID	Unique ID of Product
A1 – A3	Product Attributes (see 5.6.2)
Initial_Cost	Cost of Product
Start_Price / Initial_Price	Initial Price of Product
End_Price	Price at the End of the Simulation Run
Change	Change of Price Start to End of Simulation

Table 5.7 Variables Description

5.7.1.2 Customers

Similar to the product objects in the simulation, a set of 150 unique customers is used and assigned to the individual stores in the simulation. Using this set of customer data (characteristics) makes the stores unique (or different) in the sense that a store's demand characteristics are a result of the demand characteristics of the individual customers that shop at a particular store. Each customer is copied twice to the simulation database; one customer object for each of the stores in a pair (pMAEBI store and comparison store).

To add dynamic variation to the simulation, that is having changing demand characteristics, customers can change throughout the simulation. This is done to require that the systems (pMAEBI and "comparison system") respond to the changes in demand. It also reflects "untypical" customer behaviour (e.g. impulse buying – "I want this special chocolate now, even though it is usually too expensive"). To do so, a customer's characteristics, expressed by the attributes 1 – 3 can change. During each simulation step a random number of customers from each store is selected and the attributes of those customers are changed (A1 – A3; Section 5.6.2 described the attributes and the discrete set). If an attribute changes the attribute will be reassigned with the next higher or lower value (e.g. 0.25 becomes 0.5) of that set. This makes a customer either more "demanding" (higher value -> expects more value) or more "indifferent" (smaller value -> easier to satisfy).

5.7.2 Results

The simulation results presented here are averaged over three runs. In each run 3 pairs of stores sell 50 different products each and have a customer base of 50 shoppers. Each of the pair consists of a HQ Store (old centralised approach) and an Agent Store (pMAEBI approach) that have the same start up parameter. This means that the characteristics of the customer and the products are equal, thus a direct comparison is possible. In other words, the results present 9 comparisons between de-centralised (MAEBI) and a centralised (traditional) approach. In all scenarios all decisions/recommendations that the systems make are implemented as they are without any changes.

	Unit Sales	Total Cost	Total Revenue	Profit	Avg Price
pMAEBI Stores	617,504	204,916.25	383,668.25	178,752.00	0.62
comparison stores	1,785,407	609,400.50	732,277.25	122,876.75	0.34

Table 5.8 Results (Total)

	Avg Unit Sales	Avg Cost	Avg Revenue	Avg Profit
pMAEBI Stores	68,612	22,768.47	42,629.81	19,861.33
comparison stores	198,379	67,711.17	81,364.14	13,652.97

Table 5.9 Average results over all stores in all simulation runs

Table 5.8 shows a summary of the overall sales of all stores over 3 simulation runs. Table 5.9 summarises the same data as Table 5.8, but as averages instead of totals (all stores over 3 simulation runs grouped by store type). The results show that the pMAEBI managed stores sold fewer units (34.58% of the sales volume of the comparison group) than the comparisons stores, which explains the differences in cost and revenue of the stores. Unlike the comparison stores, the pMAEBI stores realised a much higher average sales price (82.3% over the average price of the comparison group) that resulted in higher unit margins and higher store profits. The comparison stores made a profit that was 68.74 % of the profit of the pMAEBI stores and it is due to the high unit sales. In Figure 5.13 these numbers are broken down to stores (1 – 6) and grouped by pMAEBI (y) or comparison group (n).

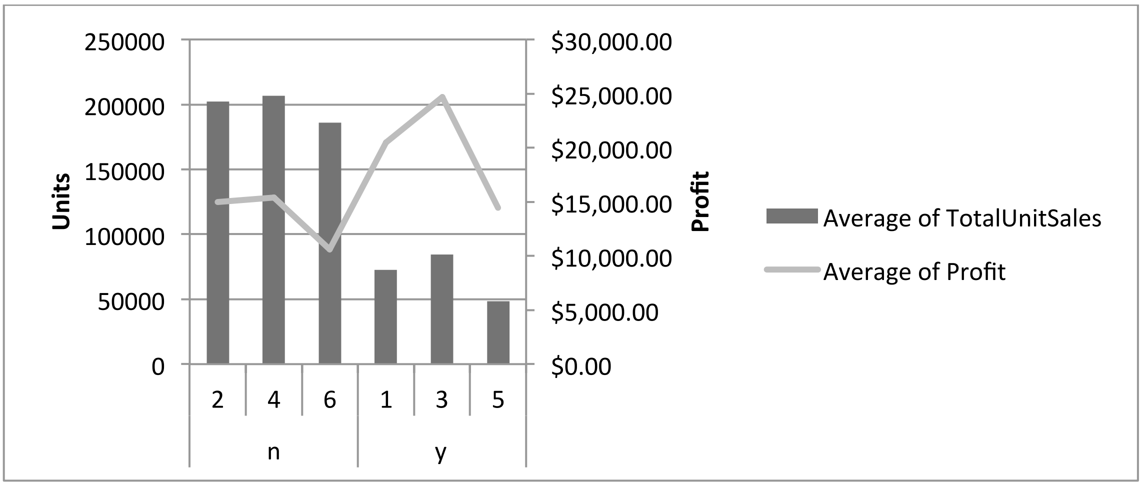

Figure 5.13 Sales / Profit

Stores that implement the new pMAEBI architecture tend to sell less for more and achieved higher margins than the 'competing' stores. This indicates that utilising the store data better captures customers' preferences.

The results of the simulations presented here suggest that the pMAEBI system (as an instance of MAEBI framework) does have advantages over the centralised / traditional system design. The results are understandably limited to the complexity of the simulation. This means it might be beneficial to extend the simulation system towards a more realistic artificial economy. This might help to evaluate in more detail other, in particular qualitative benefits of MAEBI, for example during deployment.

5.8 Chapter Summary

This chapter covered the implementation and evolution of the artifact, activities 4 and 5 in Peffers et al. (2008) research process.

The chapter first introduced retail pricing as the chosen problem domain to implement the testbed system. Pricing is a problem that is applicable to all businesses, and becomes particularly complex for retail chains that have to price many thousand of products across stores. Business literature suggests that retailers have to pay more attention to local market characteristics and include these into their decision making.

The testbed consist of three parts, a retail simulation, a centralised BI system that acts as the comparison system and the pMAEBI system that was designed based on the MAEBI concept proposed in this research.

The purpose of the simulation is to show that the MAEBI design approach shows advantages in this artificial economy. Advantage here is solely expressed as store profit. Initial results that are presented here do suggest that the approach is viable and that the store performance could be improved compared to a centralised approach.

Chapter 6 – Research Evaluation

6.1 Introduction

Hevner et al. (2004) stress the points of relevance and rigor in IS design science research (DSR) projects. This chapter summarises the research activities and results and argues those against Hevner's et al (2004) DSR guidelines. This is done to ultimately validate this research as a DSR project as described by Hevner.

The remainder of the chapter is organised around the 7 guidelines. Each guideline is addressed in terms of what was proposed and how this was translated or implemented in the research. Where applicable, key literature is revisited to better communicate the context and relevance of the research.

6.2 Hevner's DSR Guidelines

6.2.1 Design as an artifact

Hevner's et al. (2004) first guideline describes the requirement regarding the outcome of the research. They say "The result of design-science research in IS is, by definition, a purposeful IT artifact created to address an important organizational problem. It must be described effectively, enabling its implementation and application in an appropriate domain." (Hevner, et al., 2004, p. 82)

Chapter 4 introduces the MAEBI concept, the "purposeful IT artefact". The MAEBI framework was designed to advance the current BI concept and provide decision making capabilities throughout an organisation. Based on the literature in the areas of Business Intelligence (BI), Agent and Multi Agent Systems and

Management, six issues were identified where current BI concepts do not support businesses and decision makers as technology would allow. Those issues were transformed into design objectives that guided the design of the MAEBI concept. Considering that decision making is a core activity in every business, it is safe to say that this research addresses an “important organizational problem”.

The two components of the MAEBI system, the Configuration Engine (CE) and Decision Unit (DU) were described in detail, including the individual modules that comprise the respective components. Comments and suggestions about technologies that can be used to implement systems based on MAEBI are given. To further illustrate an implementation or instantiation of the MAEBI framework, the pMAEBI system adapts the concept in the retail pricing domain and shows how to leverage local knowledge to improve pricing decisions.

With respect to Hevner’s et al. (2004) guideline, the designed artefact was described in detail in this thesis in combination with a prototype implementation in a business context. This should allow an appropriately trained person to adopt the concept into an IS system. However, the presented concept is to be understood as research and not a fully functional product or architecture.

6.2.2 Problem Relevance

Guideline 2 addresses the need for relevance. “The objective of design-science research is to develop technology-based solutions to important and relevant business problems.” (Hevner, et al., 2004, p. 83)

Decision Making is at the core of every business, regardless of size, industry and location. We have witnessed a significant increase in data that is available to the decision maker but decision support systems, by and large, have not yet developed at

a similar rate. Designing new or advanced existing concepts and models, like MAEBI, that can improve business decision making by utilising different and/or new technologies does surely present "technology-based solutions to "important and relevant business problems" as Hevner calls it.
The prototype implementation in the context of retail pricing further develops the relation between the technological solution, the pMAEBI system, and the business problem, pricing, and in consequence a company's profits. Hevner et al (2004) explicitly mention that business goals and opportunities are often related to cost and/or profit and that IS systems play a major role in this context.

Generally, MAEBI based systems should add value in various business situations, in particular those where local and global objectives can differ. Rigby and Vishwanath (2006, p. 82) argue that localisation supersedes centralisation and that successful retailers have to "… cater to local differences while maintaining scale efficiencies.". Such situations are by no means limited to retail but can be found in areas like logistics / fleet management, where each car, truck or ship has access to global delivery and cost data and the decision unit can for example be enriched by local traffic and weather data. Other areas that should be considered in future research are finance, healthcare (e.g. patient monitoring) or mobile commerce.

6.2.3 Design Evaluation

The evaluation of the designed artefact is an essential part of the research process. "The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods." (Hevner, et al., 2004, p. 83). More specifically "Thus evaluation includes the integration of the artifact within the technical infrastructure of the business environment." (Hevner, et al., 2004, p. 85). To do so and evaluate the MAEBI framework, a prototype (pMAEBI) was implemented and integrated into a simulation based testbed system. Simulation is one of the appropriate evaluation methods that are suggested by Hevner in a DSR context. Simulation based evaluation is also suggested by Theodoropoulos et al. (2009) in an agent and multi agent context. The authors argue "Simulation is therefore the only viable method to rigorously study their properties …". Agent based systems, like the MAEBI concept itself and the pMAEBI prototype, are complex and often experimental systems and it "… can be difficult to formally verify their properties ..." (Theodoropoulos, et al., 2009, p. 77). The design of the simulation was adopted from literature, for example Jager (2007) who formalises the 4Ps (Price, Product, Placement , Promotion) for the specific reason of social simulation models. Neri (2007) uses a similar discrete approach to describe customer and product values.

In addition to the simulation testing, it is to be noted what Vaishnavi and Kuechler (2007, p. 24) say about continuous evaluation during the design process: "In a sense, evaluation takes place continuously in a design process (research or otherwise) because a large number of "micro-evaluations" take place at every design detail decision. Each decision is followed by a "thought experiment" in which that part of the design is mentally exercised by the designer.

6.2.4 Contribution

Hevner et al. (2004) emphasise the need for clear contribution of a DS research project and state "The ultimate assessment for any research is, 'What are the new and interesting contributions?'". Such a contribution can be in the areas of design artefact, design foundations, and/or design methodologies.

As outlined in chapter 3, the primary contribution of this research is the artefact, the MAEBI concept itself. MAEBI is the result of the purposeful combination of Business Intelligence concepts and agent / multi agent technology to better support business in their decision making tasks. In particular, decision making on a local (operation) level, an area (level) where BI systems traditionally were not used. Thus the research outcome is what Hevner et al. (2004, p. 87) describe as "… apply existing knowledge in new and innovative ways.".

Secondary contributions of this research can be found in the contribution towards the body of knowledge in the area of agent and multi agent systems. Several authors (e.g. DeLoach, 2009; Georgeff, 2009; Winikoff, 2009) have discussed the future of agent oriented software design and what the obstacles are and reasons why the approach is not as well reflected in mainstream software engineering as hoped for. For example Weyns et al. (2008) discuss the future of agent oriented software development and identified the lack of integration between multi agent systems and general purpose technologies as one of the technology obstacles of agent technology adoption. In particular the developed prototype demonstrated how such systems can be implemented on standard hardware and software (like SQL Server and SSAS). Also the pMAEBI system demonstrates an implementation where the decentralised and autonomous nature of the agent paradigm supports the overall design objectives. DeLoach (2009) comment "… it is still possible to envision a non-agent approach that is equally suited for the task.", cannot be proven wrong. However, the agent metaphor – intelligent agents embedded in the environment, work to achieve a more desirable outcome for that environment (e.g. store) – seems to suit better than other programming metaphors.

As part of the literature review and the design of the testbed it was argued that testing and evaluation of agent and multi agent systems is difficult. This difficulty is because of the complexity of such systems and the lack of established best practice methods. The testbed that was implemented to test and evaluate the pMAEBI prototype presents an instantiation of an evaluation system and might help in the process of finding some agreed methods for testing of agent based systems.

6.2.5 Research Rigor

Hevner et al. (2004, p. 87) describe the importance of rigor in DSR as "Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artefact." The authors further specify the requirement by arguing, "… rigor is derived from the effective use of the knowledge base - theoretical foundations and research methodologies." Hevner et al. (2004, p. 88).

Chapter 3 describes the methodology that was established for this research. Based on Jonker & Pennink's (2009) Research pyramid, paradigm, methodology, methods and techniques were determined and reasoned.

The underlying research process (sequence) was adopted from Peffers et al. (2008). The model is shown in Figure 6.1 and the mapping is described in Table 6.1. The entry point for this research is, "Problem Centred Initiation"; a problem that was identified in existing literature was investigated in the research process.

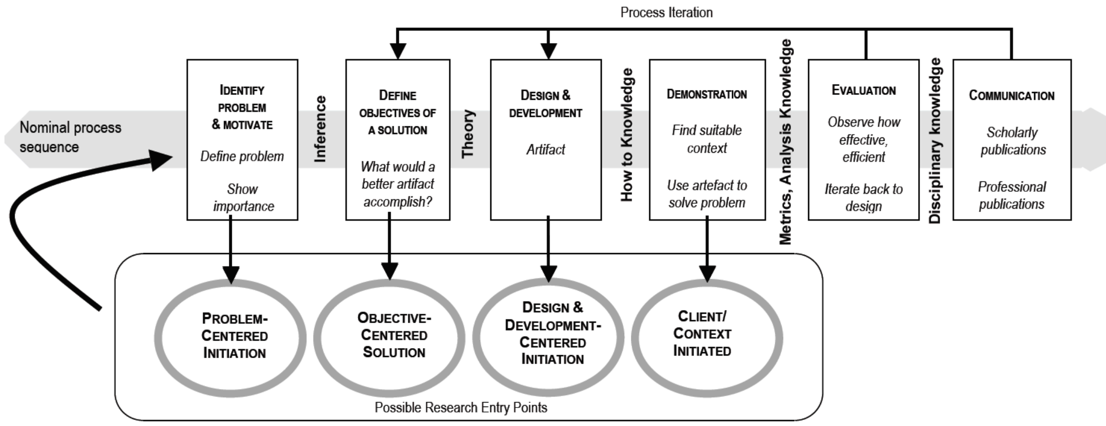

Figure 6.1 Design Science Research Methodology Process (Peffers, et al., 2008)

Peffers et al. DSRM Process	Thesis
Activity 1: **Identify Problem & Motivate**	Chapter 1 – Introduction to the Problem domain Chapter 2 – Literature Review
Activity 2: **Define Objectives of Solution**	Chapter 4 – Identification of Design Objectives
Activity 3: **Design and Development**	Chapter 4 – Design MAEBI Concept
Activity 4: **Demonstration**	Chapter 5 – pMAEBI Implementation
Activity 5: **Evaluation**	Chapter 5 - Simulation
Activity 6: **Communication**	Chapter 6 – List of Publication/Talks

Table 6.1 Peffers et al. (2008) Process mapping

Each step of the research is based on knowledge gathered from literature of the respective reference disciplines (see Ch. 2). Those reference disciplines are in particular Business Intelligence and Agent and Multi Agent technology.

Methods and techniques that were used in the context of this research, to design, implement and evaluate the artefact and the corresponding implementation are either adopted from reference literature or adapted where necessary. For example, the choice of simulation as the method of evaluation was made as both the reference literature in DSR and the reference literature in agent and multi agent systems suggest that simulation is an appropriate method (e.g. Hevner, et al., 2004; Theodoropoulos, et al., 2009) .

6.2.6 Design as a Search Process

Design science research is an iterative (search) process that aims at leveraging knowledge from reference disciplines to solve business problems.

MAEBI is the result of a search process. Current methods and systems were identified in the literature and compared against requirements and practices in business. The 6 design objectives are the "result" of that search process. To use agent technology to implement those objectives is again the result of a search process of available development methods.

6.2.7 Communication of Research

The results of design science research projects are hopefully interesting and relevant for both management and technology-oriented audiences. However, those audiences have different perspectives and information needs. Hevner et al. (2004, p. 83) write "Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences."

The research conducted and described in this thesis draws from a variety of different technologies and concepts and the primary audience are technology-oriented academics.

In addition to this thesis, the research or part of it were presented at various occasions throughout the research process.

During the initial stage of the research, a proposal was presented to a mixed audience of senior academics and PhD students at the Doctoral Consortium at the ACIS 07 conference.

On invitation of the President of the Lions Club Brisbane Central, business and managerial aspects of the research were presented to a selected audience of Australian business professionals and executives. While the idea/system was well received and the audience agreed that there is an application for such systems, no specific feedback was given that was used in the research.

The paper "A Multi-Agent Framework for Distributed Business Intelligence Systems" was presented at the 45th HICCS conference 2012.

6.3 Summary

The chapter re-visited the different parts of the research and argues those against the 7 Design Science Research guideline suggested by Hevner et al. (2004). This was done to satisfy the requirement of a research evaluation, which is part of the DSR process.
This thesis documents all aspects of the search and design process and the resulting artefact the MAEBI concept. The artefact was evaluated in a testbed environment that was built to emulate the environment the artefact has to perform in.

As the research presented here does address all 7 guidelines outlined by Hevner and knowledge is drawn from well published research, this meets the criteria for a valid design science research project.

Chapter 7 Conclusion & Future Research

7.1 Summary

This thesis documents a research project about the design of a new form of a Business Intelligence system. The motivation behind this research is a literature review that identified several "issues" where the traditional BI approach does not support business to its fullest potential. For example Edmund and Morris (2000) report on the issue of information overload in organisations or Barone et al. (2010) argue that is still difficult to make sense out of the available data. This "lack of support" is on one hand due to changes in business needs and on the other hand is related to technology.

Business has changed significantly and become more complex (e.g. competition, globalisation, etc.) The business phrase "Think Global. Act Local." is used to describe the goal of combining the efficiencies of a large organisation but still allowing for adjustment to local market characteristics. However companies in the past rather focused on the "global part" of the phrase. This has changed due to more educated and demanding customers (e.g. easier access to information) and general competitive pressure. Advances in IS/IT (e.g. barcode, RFID, Internet) has led to the situation that businesses generate and store significant amounts of data, however turning this data into actionable information and into decisions is still a challenge (e.g. Barone, et al., 2010).

The BI concept is build around a Data Warehouse (DW) that acts as a long term data repository for a variety of organisational data. It is common practice to schedule ETL (copying and cleaning) processes to run at off peak times (e.g. weekend, after business closes), which inevitably leads to out of date data (information) in the DW. Another practice is it to aggregate/summarise data in this process (e.g. daily sales to weekly sales). BI is traditionally a strategic/tactical decision support tool and "real-time" data in a transactional granularity is not necessarily required.

In recent years we have seen a shift towards BI applications in operational environments to support decision makers at that level to make decisions in these complex (e.g. amount of data, timeliness, etc.) environments. To better align BI with these business challenges, this research proposed the Multi Agent Enhanced Business Intelligence (MAEBI) concept that utilises agent and multi agent technology to encapsulate decision making functionality and distribute this functionality throughout the organisation.

7.2 Research findings

This research concerned itself with the question:

"Does the combination of Business Intelligence and Multi Agent Systems provide an advantage compared to centralised Business Intelligence in respect to its applicability to deliver localised decision automation in multi store retail organisations?"

In summary, the research results suggest that the proposed BI/MAS combination and its local decision making focus does provide an advantage (measured in store profit) compared to traditional (centralised) BI in multi store retail chains.

In more detail, to address the research question a new BI derived concept, called Multi Agent Enhanced Business Intelligence (MAEBI) was developed to 'combine' BI and MAS. The enhancement of the system in comparison to traditional/centralised BI is that MAEBI focuses on localised operational decision making instead of centralised strategic/tactical decision support. The traditional BI architecture does not well reflect this decentralisation or local view on data and analysis, as it is a centralised system. Implementations where big data warehouses are broken down into smaller cubes and data marts might help in regards to organisation but they do not change the architecture.

The core of the MAEBI framework is a so called Decision Unit (DU) that encapsulates all functionality that is required from data access to decision implementation. A DU consists of a communication, a data storage, a data analysis, a learning and a decision execution module. This means it contains components that are known from today's BI systems with the extension of the decision execution module, that allow implementation of the decision that the system has made. The second component of the MAEBI framework is a Configuration Engine (CE). It is concerned with administrative tasks in the system. For example the CE creates new DUs and provides data that is required for the initialisation process. Despite the focus on localised decision making, some central control has to be maintained and the CE reflects this.

The research questions focuses in particular on the application in a retail environment and the pricing problem. This problem domain was chosen as pricing in general, but in particular the pricing in retail chains is one of those areas where local data can be more valuable in the process than global or in some form summarised data. In pricing, issues such as local tastes and socio-economic level will influence demand for specific products, which may differ from demand in other locations.

Based on the MAEBI concept a prototype, pMAEBI (p = pricing), was designed and implemented. It is the purpose of the system to determine the sales price for a given product/store combination based on the customer characteristics of the particular store, using local data for a local analysis in the DU. The DU can access sales data of "its" store and analyse the data. Once the DU made a decision, it can alter the price of a product through the decision execution module.

To test and evaluate the prototype a testbed system was implemented, that included a retail simulation, a traditional/centralised BI system (reference system) and the pMAEBI system. A simulation based approach was chosen as this technique is suggested as a suitable technique in Hevner's et al. (2004) DSR guidelines and in agent literature. The simulation replicates a retail environment that includes stores, products and customers. Stores and products are passive elements in the simulation whereas the customers are active objects and are the source of demand. Customers base their decision on personal preferences and price to evaluate the (personal) value of each product and decide whether to buy the product or not. The traditional BI system is the reference system and replicates the "copy all to central DW" workflow. The pricing method was implemented using Microsoft SSAS and individual ANNs to reflect the product/store combinations.

The simulation results indicate that the distributed nature of pMAEBI and the local perspective of the DUs could improve store profitability compared to a traditional centralised system. This is understood as a proof of concept of the proposed MAEBI concept. Comparing the results of the two systems in the simulation suggests that the pMAEBI, using local transactional data, outperforms the comparison stores using the centralised system. The simulation results are much to be expected, however their real role was to provide a test environment to show that the architecture was in fact a viable solution (design).

The contribution of this research is primarily to the field of DSS/BI by proposing a BI system design that uses agents to encapsulate BI (decision making) functionality/capability and distribute those throughout an organisation. This localised focus is different to traditionally centralised approach of BI and allows to better utilisation of available data for decision making.

pMAEBI presents an implemented instance of the MAEBI architecture and illustrates how the research findings can be used in a practical (business) context. In general the architecture could allow business to improve their decision making especially in areas where local characteristics differ.

The aim of the research is to show that the MAEBI is a viable solution. Using synthetic data obviously is a limiting factor as not all of the complexity of real data/real problem domain is represented. While the concept aims at many different decision environments, a proof of concept was only investigated in one area (retail pricing). These limitations are interesting opportunities for future research that are presented in the next section.

7.3 Future Research

The research presented here focuses on different technologies and concepts that present interesting research opportunities for the future. Decision Support systems in all their variations are an ongoing research area and agent and multi agent systems will require more research efforts to gain broader adaptation.

There are in particular opportunities in the areas of testing, implementation and application.

The testing and evaluation of the system was done in an artificial simulation environment and is a limiting factor in the research. Obviously more testing in different environments is required to further formalise and optimise the concept. Simulation seems to be an appropriate method, however more complex simulation systems might provide additional insights in the performance of the MAEBI concept. The simulation could be improved by implementing more complex customer behaviour models and bringing more decision variables/characteristics into the process. It is probably interesting to expand the simulation to a "virtual economy" with many sellers and buyers. Ultimately however an implementation with an industry partner is the most desirable evaluation method and may provide some practical insight that was not yet considered in literature.

An interesting part of this work was the evaluation of Axum as a potential agent environment. Axum and in particular the integration with mainstream development tools, is an interesting and promising approach to allow the use of agent oriented software development and system design on a broader scale. Although Microsoft is not continuing with Axum as a product, a number of the concepts will undoubtedly become further developed. MAEBI itself is not bound to a specific system/programing language and a formal evaluation of different implementation strategies might increase applicability in different environments.

The pMAEBI system focuses exclusively on the pricing problem, however the MAEBI architecture is likely to be relevant in different contexts. In particular it will be relevant in domains with high decision frequencies and where local characteristics differ. Fleet management or patient monitoring are examples of such environments. This should be further explored by implementing (test) systems in such areas and again formalise decision criteria when/how a MAEBI system is an appropriate choice.

This research focused on the combination of agent technology in context of business intelligence systems to address issues that were identified in literature were the current centralised BI approach does not provide an optimal fit. Using a design science research approach, the MAEBI concept was developed and a proof of concepts system and test-bed was implemented in a retail/pricing context. The simulation showed that the MAEBI managed stores performed better. Future research should focus on more complex data/problems and different areas of application.

Bibliography

Alon, I., Qi, M., & Sadowski, R. J. (2001). Forecasting aggregate retail sales:: a comparison of artificial neural networks and traditional methods. [doi: DOI: 10.1016/S0969-6989(00)00011-4]. *Journal of Retailing and Consumer Services, 8*(3), 147-156.

Anderson-Lehman, R., Watson, H. J., Wixom, B. H., & Hoffer, J. A. (2008). Flying High with Real-Time Business Intelligence *Handbook on Decision Support Systems 2* (pp. 443-462).

Anthony, R. N. (1965). *Planning and Control Systems: A Framework for Analysis*: Division of Research, Graduate School of Business Administration, Harvard University.

Arnott, D., & Pervan, G. (2008). Eight key issues for the decision support systems discipline. *Decision Support Systems, 44*(3), 657-672.

Azvine, B., Cui, Z., Majeed, B., & Spott, M. (2007). Operational risk management with real-time business intelligence. *BT Technology Journal, 25*(1), 154-167.

Azvine, B., Cui, Z., & Nauck, D. D. (2005). Towards real-time business intelligence. *BT Technology Journal, 23*(3), 214-225.

Azvine, B., Cui, Z., Nauck, D. D., & Majeed, B. (2006). *Real Time Business Intelligence for the Adaptive Enterprise.* Paper presented at the E-Commerce Technology, 2006. The 8th IEEE International Conference on and Enterprise Computing, E-Commerce, and E-Services, The 3rd IEEE International Conference on.

Barone, D., Yu, E., Won, J., Jiang, L., & Mylopoulos, J. (2010). Enterprise Modeling for Business Intelligence. In P. Bommel, S. Hoppenbrouwers, S. Overbeek, E. Proper & J. Barjis (Eds.), *The Practice of Enterprise Modeling* (Vol. 68, pp. 31-45): Springer Berlin Heidelberg.

Baydar, C. (2003). Agent-based modeling and simulation of store performance for personalized pricing. *Simulation Conference, 2003. Proceedings of the 2003 Winter, 2*.

Baydar, C. (2008). Optimization of Store Performance Using Personalized Pricing *Evolutionary Computation in Practice* (pp. 143-161).

Bellifemine, F., Caire, G., & Greenwood, D. (2007). Developing multi-agent systems with JADE.

Bolton, R. N., Shankar, V., & Montoya, D. (2005). Recend Trends and Emerging Practices in Retailer Pricing. In M. Krafft & M. K. Mantrala (Eds.), *Retailing in the 21st Century* (pp. 255-269). Berlin / Heidelberg Germany: Springer.

Boyd, A. (2007). *The Future of Pricing: How Airline Ticket Pricing Has Inspired a Revolution* (1st ed.): Palgrave Macmillan.

Braubach, L., Pokahr, A., & Lamersdorf, W. (2006). Tools and Standards *Multiagent Engineering: Theory and Applications in Enterprises* (pp. 504-530): Springer.

Breath, C. M., & Ives, B. (1986). Competitive Information Systems in Support of Pricing. *MIS Quarterly, 10*(1), 85-96.

Bucklin, R., Lehmann, D., & Little, J. (1998). From Decision Support to Decision Automation: A 2020 Vision. *Marketing Letters, 9*(3), 235-246.

Burstein, F., & Holsapple, C. (2008). Preface. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 2* (pp. VI-XIX): Springer Berlin Heidelberg.

Cao, L. (Ed.). (2009). *Introduction to Agent Mining Interaction and Integration*: Springer US.

Cao, L., Luo, C., & Zhang, C. (2007). Agent-Mining Interaction: An Emerging Area? *Lecture Notes in Computer Science, 4476*, 60-73.

Carlsson, S. A., & Sawy, O. A. (2008). Decision Support in Turbulent and High-Velocity Environments

Handbook on Decision Support Systems 2. In F. Burstein & C. Holsapple (Eds.), (pp. 3-17): Springer Berlin Heidelberg.

Cassaigne, N., & Lorimier, L. (2006). A Challenging Future for i-DMSS *Intelligent Decision-making Support Systems* (pp. 401-422).

Cobuild, C. (Ed.) (1987) English language dictionary. London: Harper Collins.

Cummings, M. L. (2004). Automation Bias in Intelligent Time Critical Decision Support Systems. *AIAA Intelligent Systems, Chicago, IL*.

D'Souza, E., & White, E. (2006). Demand Forecasting for the Net Age: From Thought to Fulfillment in One Click. In Y. Lan & B. Unhelkar (Eds.), *Global Integrated Supply Chain Systems*: IGI Global.

Davenport, T. H. (2006). Competing on Analytics. [Article]. *Harvard Business Review, 84*(1), 98-107.

Davenport, T. H., & Harris, J. G. (2007). *Competing on Analytics: The New Science of Winning* (1st ed.): Harvard Business School Press.

DeLoach, S. A. (2009). Moving multi-agent systems from research to practice. *International Journal of Agent-Oriented Software Engineering, 3*(4), 5. doi: 10.1504/IJAOSE.2009.025315

Denning, P. J. (1997). A new social contract for research. *Commun. ACM, 40*(2), 132-134. doi: 10.1145/253671.253755

Dixit, A., Whipple, T. W., Zinkhan, G. M., & Gailey, E. (2007). A taxonomy of information technology-enhanced pricing strategies. *Journal of Business Research, 61*(4), 275-283.

Dolgui, A., & Proth, J.-M. (2010). Pricing strategies and models. [doi: DOI: 10.1016/j.arcontrol.2010.02.005]. *Annual Reviews in Control, 34*(1), 101-110.

Drucker, P. F. (1993). *Managing for the Future: The 1990s and Beyond*: Plume.

Edmunds, A., & Morris, A. (2000). The problem of information overload in business organisations: a review of the literature. *International Journal of Information Management, 20*(1), 17-28. doi: 10.1016/s0268-4012(99)00051-1

Efraim, T., Aronson , J. E., & Liang, T. P. (2005). *Decision Support Systems and Intelligent Systems* (7th ed.): Prentice Hall.

Elmaghraby, W., & Keskinocak, P. (2003). Dynamic Pricing in the Presence of Inventory Considerations: Research Overview, Current Practices, and Future Directions. *Management Science, 49*(10), 1287-1309.

Fisher, M. L., Raman, A., & McClelland, A. S. (2000). Rocket Science Retailing Is Almost Here: Are You Ready? *Harvard Business Review, 78*(4), 115-124.

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). *Artificial Intelligence Through Simulated Evolution*: John Wiley & Sons.

Fox, E. J., & Sethuraman, R. (2006). Retail Competition. In M. Krafft & M. K. Mantrala (Eds.), *Retailing in the 21st Century* (pp. 193-208): Springer Berlin Heidelberg.

Galliers, R. D., & Land, F. F. (1987). Viewpoint: choosing appropriate information systems research methodologies. *Commun. ACM, 30*(11), 901-902. doi: 10.1145/32206.315753

Gallupe, R. B. (2007). The tyranny of methodologies in information systems research. *The DATA BASE for Advances in Information Sytems, 38*(3), 20-28.

Georgeff, M. (2009). The gap between software engineering and multi-agent systems: bridging the divide. *International Journal of Agent-Oriented Software Engineering, 3*(4), 391-396. doi: 10.1504/IJAOSE.2009.025317

Gorry, G. A., & Scott Morton, M. S. (1971). *A Framework for Management Information Systems*: Massachusetts Institute of Technology.

Gregor, S. (2006). The Nature of Theory in Information Systems. *MIS Quarterly, 30*(3), 611-642.

Gregor, S., & Hevner, A. (2011). (Working Paper) Positioning and presenting design science research for maximum impact.

Grewal, D., Levy, M., & Kumar, V. (2009). Customer Experience Management in Retailing: An Organizing Framework. *Journal of Retailing, 85*(1), 1-14.

Gummesson, E. (1999). *Qualitative Methods in Management Research* (2nd ed.): Sage Publications.

Gustafsson, N. (2009a). Axum Language Overview Retrieved from http://www.microsoft.com/downloads/en/details.aspx?FamilyID=cfe70d5d-37aa-4c4c-8eeb-d4576c41baa2&displayLang=en

Gustafsson, N. (2009b). Axum. A .NET Language for Safe and Scalable Concurrency. *Microsoft PDC 09*.

Haas, M. W., Mills, R. F., & Grimaila, M. R. (2011). Aiding Understanding of a Contested Information Environment‚Äôs Effect on Operations

Human-in-the-Loop Simulations. In L. Rothrock & S. Narayanan (Eds.), (pp. 175-202): Springer London.

Hall, D. J. (2008). Decision Makers and Their Need for Support. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 83-102): Springer Berlin Heidelberg.

Hanks, S., Pollack, M. E., & Cohen, P. R. (1993). Benchmarks, Test Beds, Controlled Experimentation, and the Design of Agent Architectures. *AI Magazine, 14*(4), 17-42.

Helleboogh, A., Weyns, D., & Holvoet, T. (2009). On the Role of Software Architecture for Simulating Multi-Agent Systems. In A. M. Uhrmacher & D. Weyns (Eds.), *Multi-Agent Systems: Simulation and Applications*: CRC Press.

Herrler, R., & Kluegl, F. (2006). Simulation. In S. Krin, O. Herzog, P. Lockermann & O. Spaniol (Eds.), *Multi Agent Engineering* (pp. 576-596). Berlin, Heidelber: Springer.

Hess, T. J., Rees, L. P., & Rakes, T. R. (2008). Using Autonomous Software Agents in Decision Support Systems. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 529-555): Springer Berlin Heidelberg.

Hevner, A. R., & Chatterjee, S. (2010). *Design Research in Information Systems: Theory and Practice* (1st ed.): Springer.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly, 28*(1), 75-105.

Himmelspach, J., Rohl, M., & Uhrmacher, A. M. (2003). *Simulation for testing software agents - An exploration based on James*. New York: IEEE.

Holsapple, C. (2008a). Decisions and Knowledge. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 21-53): Springer Berlin Heidelberg.

Holsapple, C. (2008b). DSS Architecture and Types. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 163-189): Springer Berlin Heidelberg.

Holsapple, C., Jacob, V. S., Pakath, R., & Zaveri, J. S. (2008). Adaptive Decision Support Systems via Problem Processor Learning. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 659-696): Springer Berlin Heidelberg.

Imam, I. F., & Kodratoff, Y. (1997). Intelligent Adaptive Agents. *AI Magazine, 18*(2), 75-80.

Jager, W. (2007). The four P's in social simulation, a perspective on how marketing could benefit from the use of social simulation. [doi: DOI: 10.1016/j.jbusres.2007.02.003]. *Journal of Business Research, 60*(8), 868-875.

Janus, P., & Fouche, G. (2009). *Pro SQL Server 2008 Analysis Services*: Apress.

Jonker, J., & Pennink, B. (2009). *The Essence of Research Methodology*: Springer Berlin Heidelberg.

Kemper, H.-G., & Baars, H. (2009). *From data warehouses to transformation hubs - A conceptual architecture.* Paper presented at the European Conference on Information Systems (ECIS), Verona, Italy.

Khan, S., Ganguly, A. R., & Gupta, A. (2008). Data Mining and Data Fusion for Enhanced Decision Support. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 581-608): Springer Berlin Heidelberg.

Kimball, R., & Ross, M. (2002). *The Data Warehouse Toolkit* (second ed.): John Wiley & Sons, Inc.

Kirn, S. (2006). Flexibility of Multiagent Systems *Multiagent Engineering* (pp. 53-69).

Knapik, M., & Johnson, J. B. (1997). *Developing Intelligent Agents for Distributed Systems: Exploring Architectures, Techniques, and Applications* (1st ed.): Osborne/McGraw-Hill.

Kuhn, T. (1996). *The Structure of Scientific Revolutions*. Chicago: University of Chicago Press.

Lavbic, D., Rupnik, R., Bajec, M., & Krisper, M. (2007). Agent Oriented Approach for Integration of BI Systems. *Information Technology Interfaces, 2007. ITI 2007. 29th International Conference on*, 133-138.

Leavitt, N. (2010). Will NoSQL Databases Live Up to Their Promise? *Computer, 43*(2), 12-14. doi: 10.1109/MC.2010.58

Levy, D., Dutta, S., Bergen, M., & Venable, R. (1998). Price Adjustment at Multiproduct Retailers. *Managerial and Decision Economics, 19*(2), 81-120.

Levy, M., Grewal, D., Kopalle, P. K., & Hess, J. D. (2004). Emerging trends in retail pricing practice: implications for research. *Journal of Retailing, 80*(3), xiii-xxi.

Lim, C. P., & Jain, L. C. (2010). Advances in Intelligent Decision Making. In L. C. Jain & C. P. Lim (Eds.), *Handbook on Decision Making* (Vol. 4, pp. 3-28): Springer Berlin Heidelberg.

Lockemann, P. C. (2006). Agents. In S. Kirn, O. Herzog, P. Lockemann & O. Spaniol (Eds.), *Multiagent Engineering* (pp. 17-33): Springer Berlin Heidelberg.

Mantrala, M. K., & Krafft, M. (2006). *Retailing in the 21st Century: Current and Future Trends*: Springer.

March, S. T., & Storey, V. C. (2008). DESIGN SCIENCE IN THE INFORMATION SYSTEMS DISCIPLINE: AN INTRODUCTION TO THE SPECIAL ISSUE ON DESIGN SCIENCE RESEARCH. *MIS Quarterly, 32*(4), 725-730.

Marjanovic, O. (2007). The Next Stage of Operational Business Intelligence: Creating New Challenges for Business Process Management. *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, 215c-215c.

Marn, M. V., & Rosiello, R. L. (1992). Managing price, gaining profit. *McKinsey Quarterly*(4), 18-37.

Marsden, J. R. (2008). The Internet and DSS – Massive, Real-Time Data Availability Is Changing the DSS Landscape. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 2* (pp. 687-697): Springer Berlin Heidelberg.

McIntyre, S. H., & Miller, C. M. (1999). The selection and pricing of retail assortments: an empirical approach. [doi: DOI: 10.1016/S0022-4359(99)00010-X]. *Journal of Retailing, 75*(3), 295-318.

Mercer, D. (1996). *Marketing* (2nd ed.): Blackwell Publishing.

Meredith, R., O'Donnell, P., & Arnott, D. (2008). Databases and Data Warehouses for Decision Support. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 207-230): Springer Berlin Heidelberg.

Merriam-Webster. (Ed.) (2011) Merriam-Webster.

Michalewicz, Z., & Michalewicz, M. (2008). Machine intelligence, adaptive business intelligence, and natural intelligence [Research Frontier]. *Computational Intelligence Magazine, IEEE, 3*(1), 54-63.

Michalewicz, Z., Schmidt, M., Michalewicz, M., & Chiriac, C. (2007). *Adaptive Business Intelligence*: Springer.

Microsoft. Axum Programmer's Guide.

Montgomery, A. L. (2005). The Implementation Challenge of Pricing Decision Support Systems for Retail Managers. *Applied Stochastic Models in Business and Industry, 27*(4-5), 367-378.

Moss, L., & Atre, S. (2003). *Business Intelligence Roadmap*: Addison Wesley.

Natter, M., Reutterer, T., Mild, A., & Taudes, A. (2007). An Assortmentwide Decision-Support System for Dynamic Pricing and Promotion Planning in DIY Retailing. *Marketing Science, 26*(4), 576-583.

Negash, S., & Gray, P. (2008). Business Intelligence *Handbook on Decision Support Systems 2* (pp. 175-193): Springer Berlin Heidelberg.

Neri, F. (2007). Using an agent based simulation to evaluate scenarios in customers' buying behaviour *Emergent Intelligence of Networked Agents* (pp. 177-188).

Nguyen, T. M., Schiefer, J., & Tjoa, A. M. (2005). Sense & response service architecture (SARESA): an approach towards a real-time business intelligence solution and its use for a fraud detection application. *Proceedings of the 8th ACM international workshop on Data warehousing and OLAP*, 77-86.

Nguyen, T. M., & Tjoa, A. M. (2006, Feb. 12-16, 2006). *Zero-latency data warehousing (ZLDWH): the state-of-the-art and experimental implementation approaches.* Paper presented at the Research, Innovation and Vision for the Future, 2006 International Conference on.

Nunamaker, J., Chen, M., & Purdin, T. (1990-91). Systems development in information systems research. *J. Manage. Inf. Syst., 7*(3), 89-106. doi: citeulike-article-id:3469130

O'Leary, D. E. (2008). Decision Support System Evolution: Predicting, Facilitating, and Managing Knowledge Evolution. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 2* (pp. 345-367): Springer Berlin Heidelberg.

Offermann, P., Levina, O., Sch\, M., \#246, nherr, & Bub, U. (2009). *Outline of a design science research process*. Paper presented at the Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology, Philadelphia, Pennsylvania.

Padgham, L., & Winikoff, M. (2005). *Developing Intelligent Agent Systems* (1 ed.): John Wiley & Sons, Ltd.

Parasuraman, R., & Sheridan, T. B. (2000). A Model for Types and Levels of Human Interaction with Automation. [Article]. *IEEE Transactions on Systems, Man & Cybernetics: Part A, 30*(3), 286.

Park, B. H., & Kargupta, H. (2002). Distributed data mining: Algorithms, systems, and applications. *Data Mining Handbook*, 341–358.

Parsons, L. J., & Dixit, A. (2003). Using Artificial Neural Networks to Forecast Market Response. In G. P. Zhang (Ed.), Neural Networks in Business Forecasting (pp. 23-46). Retrieved from http://www.igi-global.com/chapter/neural-networks-business-forecasting/27243. doi: 10.4018/978-1-59140-176-6.ch002

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2008). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems, 24*(3), 45-77.

Peng, Y., Kou, G., Shi, Y., & Chen, Z. (2006). *A Systemic Framework for the Field of Data Mining and Knowledge Discovery.* Paper presented at the Sixth IEEE International Conference on Data Mining - Workshops (ICDMW'06).

Phillips, R. (2005). *Pricing and Revenue Optimization* (1st ed.): Stanford Business Books.

Phillips-Wren, G., & Jain, L. (2007). *Recent advances in intelligent decision technologies*. Paper presented at the Proceedings of the 11th international conference, KES 2007 and XVII Italian workshop on neural networks conference on Knowledge-based intelligent information and engineering systems: Part I, Vietri sul Mare, Italy.

Pick, R. A. (2008). Benefits of Decision Support Systems. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 719-730): Springer Berlin Heidelberg.

Pokahr, A., & Braubach, L. (2009). A Survey of Agent-oriented Development Tools. In A. El Fallah Seghrouchni, J. Dix, M. Dastani & R. H. Bordini (Eds.), *Multi-Agent Programming:* (pp. 289-329): Springer US.

Pokahr, A., Braubach, L., & Lamersdorf, W. (2005). Jadex: A BDI reasoning engine. *Multi-Agent Programming: Languages, Platforms and Applications. Springer, Berlin*.

Poutakidis, D., Winikoff, M., Padgham, L., & Zhang, Z. (2009). Debugging and Testing of Multi-Agent Systems using Design Artefacts. In A. El Fallah Seghrouchni, J. Dix, M. Dastani & R. H. Bordini (Eds.), *Multi-Agent Programming:* (pp. 215-258): Springer US.

Rabelo, R. J., & Klen, A. A. P. (2002). *Business Intelligence Support for Supply Chain Management*. Paper presented at the Proceedings of the IFIP TC5/WG5.3 Fifth IFIP/IEEE International Conference on Information Technology for Balanced Automation Systems in Manufacturing and Services: Knowledge and Technology Integration in Production and Services: Balancing Knowledge in Product and Service Life Cycle.

Rao, V. R. (1984). Pricing Research in Marketing: The State of the Art. *The Journal of Business, 57*(1), 39-60.

Ravi, V., Raman, K., & Mantrala, M. K. (2010). Applications of Intelligent Technologies in Retail Marketing. In M. Krafft & M. K. Mantrala (Eds.), *Retailing in the 21st Century* (pp. 173-187): Springer Berlin Heidelberg.

Rigby, D. K., & Vishwanath, V. (2006). Localization the Revolution in Consumer Markets. *Harvard Business Review, 84*(4), 82-92.

Russell, S. J., Norvig, P., Canny, J. F., Malik, J., & Edwards, D. D. (2002). *Artificial intelligence: A modern approach* (2nd ed.). Englewood Cliffs, NJ: Prentice Hall.

Sargut, G., & McGrath, R. G. (2011). Learning To Live with Complexity. [Article]. *Harvard Business Review, 89*(9), 68-76.

Schulte, R. W. (1998). Introducing the Zero-Latency Enterprise: Gartner Group.

Schwind, M. (2007). *Dynamic Pricing and Automated Resource Allocation for Complex Information Services: Reinforcement Learning and Combinatorial Auctions (Lecture Notes in Economics and Mathematical Systems)*: Springer-Verlag New York, Inc. Secaucus, NJ, USA.

Simon, H. (1989). *Price management*. Amsterdam, Netherlands: North-Holland.

Simon, H., & Dolan, R. J. (1996). *Power pricing: how managing price transforms the bottom line*: Free Press.

Simon, H. A. (1960). *The New Science of Management Decision*. New York: Harper Brothers.

Simon, H. A. (1996). *The Sciences of the Artificial*: MIT Press.

Sudeikat, J., Braubach, L., Pokahr, A., & Lamersdorf, W. (2005). Evaluation of Agent–Oriented Software Methodologies – Examination of the Gap Between Modeling and Platform. In J. Odell, P. Giorgini & J. Müller (Eds.), *Agent-Oriented Software Engineering V* (Vol. 3382, pp. 126-141): Springer Berlin / Heidelberg.

Symeonidis, A. L., & Mitkas, P. A. (2005). Intelligent Agents and Multi-Agent Systems *Agent Intelligence Through Data Mining* (Vol. 14, pp. 41-55): Springer US.

Tan, P.-N., Steinbach, M., & Kumar, V. (2005). *Introduction to Data Mining* (1 ed.): Addison Wesley.

Theodoropoulos, G. K., Minson, R., Ewald, R., & Lees, M. (2009). Simulation Engines for Multi-Agent Systems. In A. M. Uhrmacher & D. Weyns (Eds.), Multi-Agent Systems: Simulation and Applications (pp. 77-105): CRC Press. Retrieved from http://www.crcpress.com/product/isbn/9781420070231.

Timm, I., Scholz, T., & Fürstenau, H. (2006). From Testing to Theorem Proving. In S. Kirn, O. Herzog, P. Lockemann & O. Spaniol (Eds.), *Multiagent Engineering* (pp. 531-554): Springer Berlin Heidelberg.

Timm, I., Scholz, T., Herzog, O., Krempels, K.-H., & Spaniol, O. (2006). From Agents to Multiagent Systems. In S. Kirn, O. Herzog, P. Lockemann & O. Spaniol (Eds.), *Multiagent Engineering: Theory and Applications in Enterprises* (pp. 35-51): Springer.

Trivedi, M. (2011). Regional and Categorical Patterns in Consumer Behavior: Revealing Trends. [doi: DOI: 10.1016/j.jretai.2010.11.002]. *Journal of Retailing, 87*(1), 18-30.

Tsichritzis, D. (1997). The Dynamics of Innovation *Beyond Calculation: The Next Fifty Years of Computing* (pp. 259-265): Copernicus.

Vahidov, R., & Kersten, G. E. (2004). Decision station: situating decision support systems. [doi: DOI: 10.1016/S0167-9236(03)00099-X]. *Decision Support Systems, 38*(2), 283-303.

Vaishnavi, V. K., & Kuechler, W. J. (2007). *Design Science Research Methods and Patterns* (first ed.): Auerbach Publications.

Vercellis, C. (2009). *Business Intelligence*: John Wiley & Sons, Ltd.

Vitolo, T. M., & Coulston, C. (2004). Simulation in IS Research: Technique Underrepresented in the Field. In M. E. Whitman & A. B. Woszczynski (Eds.), *The Handbook of Information Systems Research*: Idea Group Pub.

von der Gathen, A., Daus, P. W., & Simon, H. (2005). Retail Pricing - Higher Profits Through Improved Pricing Process. In M. Krafft & M. K. Mantrala (Eds.), *Retailing in the 21st Century* (pp. 271-288). Berlin / Heidelberg Germany: Springer.

Weyns, D., Parunak, H. V. D., & Shehory, O. (2008). The Future of Software Engineering and Multi-Agent Systems. *International Journal of Agent-Oriented Software Engineering (IJAOSE)*, 1-8.

Winikoff, M. (2009). Future directions for agent-based software engineering. *International Journal of Agent-Oriented Software Engineering, 3*(4), 8. doi: 10.1504/IJAOSE.2009.025319

Wooldridge, M. (2001). *Introduction to Multiagent Systems*: John Wiley & Sons, Inc. New York, NY, USA.

Zenobia, B., Weber, C., & Daim, T. (2009). Artificial markets: A review and assessment of a new venue for innovation research. *Technovation, 29*(5), 338-350. doi: 10.1016/j.technovation.2008.09.002

Zentes, J., Morschett, D., & Schramm-Klein, H. (2007). *Strategic Retail Management* (1st ed.): Gabler.

Zhang, C., Zhang, Z., & Cao, L. (2005). Agents and Data Mining: Mutual Enhancement by Integration. *Autonomous Intelligent Systems: Agents and Data Mining, 3505*.

Zhang, Z., & Zhang, C. (2004). *Agent-Based Hybrid Intelligent Systems: An Agent-Based Framework for Complex Problem Solving*: Springer.

Zimmermann, R., Winkler, S., & Bodendorf, F. (2006). Supply Chain Event Management With Software Agents. In S. Kirn, O. Herzog, P. Lockemann & O. Spaniol (Eds.), *Multiagent Engineering* (pp. 157-175): Springer Berlin Heidelberg.

Zöller, A., Rothlauf, F., Paulussen, T. O., & Heinzl, A. (2006). Benchmarking of Multiagent Systems. In S. Krin, O. Herzog, P. Lockermann & O. Spaniol (Eds.), *Multi Agent Engineering* (pp. 557-574). Berlin, Heidelber: Springer.